دانشگاه صنعتی شریف
دانشکده مهندسی کامپیوتر

آزمایشگاه امنیت داده و شبکه
http://dnsl.ce.sharif.edu

درس ۱۰: SSL و TLS

محمد صادق دوستی

فهرست مطالب

- معرفی و تاریخچه
- SSL/TLS در سطح بالا
- TLS در عمل
- جزئیات TLS
- Heartbleed

معرفی کلی

- **SSL:** Secure Sockets Layer
- **TLS:** Transport Layer Security
- SSL در شرکت Netscape Communications توسعه یافت و به سرعت محبوب شد (طاهر الجمل؛ پدر SSL).
  - هدف اصلی SSL، امنیت وب (HTTP) بود.
  - ترکیب HTTP روی SSL را HTTPS گوییم.
  - امروزه SSL کاربردهای دیگری نیز دارد.
- TLS نسخه استاندارد شده SSL است.

جایگاه در شبکه

- SSL/TLS لایه‌ای بالای لایه انتقال در پشته پروتکل TCP/IP است.

☜ برخی آن را در زمره لایه کاربرد محسوب می‌کنند.

- SSL/TLS بر مبنای پروتکل TCP است.

☜ نسخه‌ای بر مبنای UDP هم پیاده شده است که به آن Datagram Transport Layer Security (یا DTLS) می‌گویند.

- پروتکلهایی نظیر HTTP، FTP، SMTP، NNTP و XMPP قادرند از SSL/TLS استفاده کنند.

پورتهای پیش‌فرض معروف

پروتکل	پورت عادی	پورت روی SSL/TLS
HTTP	۸۰	۴۴۳
XMPP	۸۰	۴۴۳
SMTP	۲۵ و ۵۸۷	۴۶۵
NNTP	۱۱۹	۵۶۳
FTP	۲۰ و ۲۱	۹۸۹ و ۹۹۰
IMAP	۱۴۳	۹۹۳
POP3	۱۱۰	۹۹۵
LDAP	۳۸۹	۶۳۶
Telnet	۲۳	۹۹۲

توجه: پروتکل Telnet روی SSL/TLS کاملاً با پروتکل SSH تفاوت دارد.

فرمان STARTTLS

- فرمان STARTTLS افزونه‌ای بر پروتکلهای متن آشکار است، که با اجرای آن می‌توانند امنیت خود را به کمک TLS ارتقا دهند. **مثال:** SMTP:

```
S: <waits for connection on TCP port 25>
C: <opens connection>
S: 220 mail.example.org ESMTP service ready
C: EHLO client.example.org
S: 250-mail.example.org offers welcome
S: 250 STARTTLS
C: STARTTLS
S: 220 Go ahead
C: <starts TLS negotiation>
C & S: <negotiate a TLS session>
C & S: <check result of negotiation>
C: EHLO client.example.org
```

تاریخچه

پروتکل	سال	توضیح
SSL 1.0	؟؟	داخلی Netscape – منتشر نشد – به شدت ناامن
SSL 2.0	۱۹۹۵	تعدادی ناامنی – از ۲۰۱۱ به بعد منسوخ محسوب می‌شود (RFC 6176)
SSL 3.0	۱۹۹۶	حمله POODLE به آن وارد است - از ۲۰۱۵ به بعد منسوخ محسوب می‌شود (RFC 7568)
TLS 1.0	۱۹۹۹	بر مبنای SSL 3.0 – قابلیت تنزل اتصال به SSL 3.0 و در نتیجه ناامنی
TLS 1.1	۲۰۰۶	رفع تعدادی از ناامنی‌های TLS 1.0
TLS 1.2	۲۰۰۸	افزودن برخی الگوریتمهای رمز به TLS 1.1 – عدم سازگاری با SSL 2.0
TLS 1.3	به زودی	حذف برخی الگوریتمهای رمز ضعیف – افزودن الگوریتمهای رمز جدید

فهرست مطالب

- معرفی و تاریخچه
- **SSL/TLS در سطح بالا**
- TLS در عمل
- جزئیات TLS
- Heartbleed

دو مفهوم اساسی SSL/TLS

- **نشست (Session):** تناظری بین کارخواه و کارگزار.
  - **ایده:** پارامترهای رمزنگاری (از جمله کلید نشست) یک بار تبادل شوند و پس از آن بتوان با خیال راحت انواع ارتباط را داشت.
  - **علت:** تبادل پارامترهای رمزنگاری هزینه زیادی دارد.
- **اتصال:** ارتباطی برای انتقال بسته‌ها بین کارخواه و کارگزار.
  - روی یک نشست می‌توان چندین اتصال داشت.
  - اتصالها نیاز به تبادل پارامترهای رمزنگاری ندارند و از پارامترهای نشست بهره می‌گیرند.

زیر پروتکلهای SSL/TLS

- پروتکل SSL/TLS شامل چند زیر پروتکل است:

- Record Protocol (رکورد)
- Handshake Protocol (دستداد)
- Change Cipher Spec Protocol (تغییر رمز)
- Alert Protocol (هشدار)

زیر پروتکلهای دستداد و رکورد

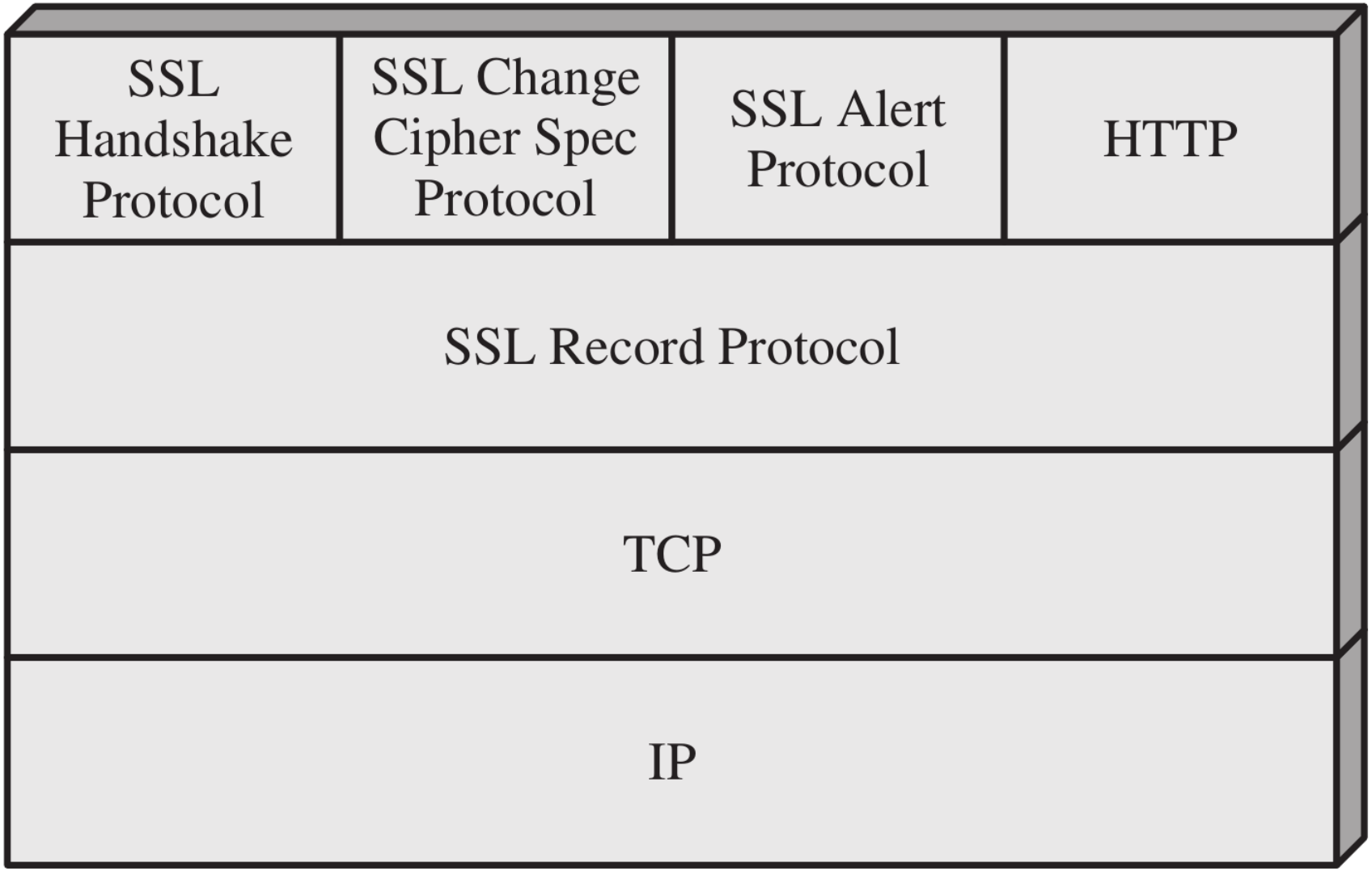
- کارخواه و کارگزار با استفاده از زیر پروتکل دستداد پارامترهای رمزنگاری را تبادل می‌کنند.

- زیر پروتکل رکورد، از پارامترهای رمزنگاری استفاده کرده و برای سایر زیر پروتکلهای SSL/TLS و پروتکل لایه کاربرد روی آن خدمات محرمانگی و صحت را فراهم می‌آورد.

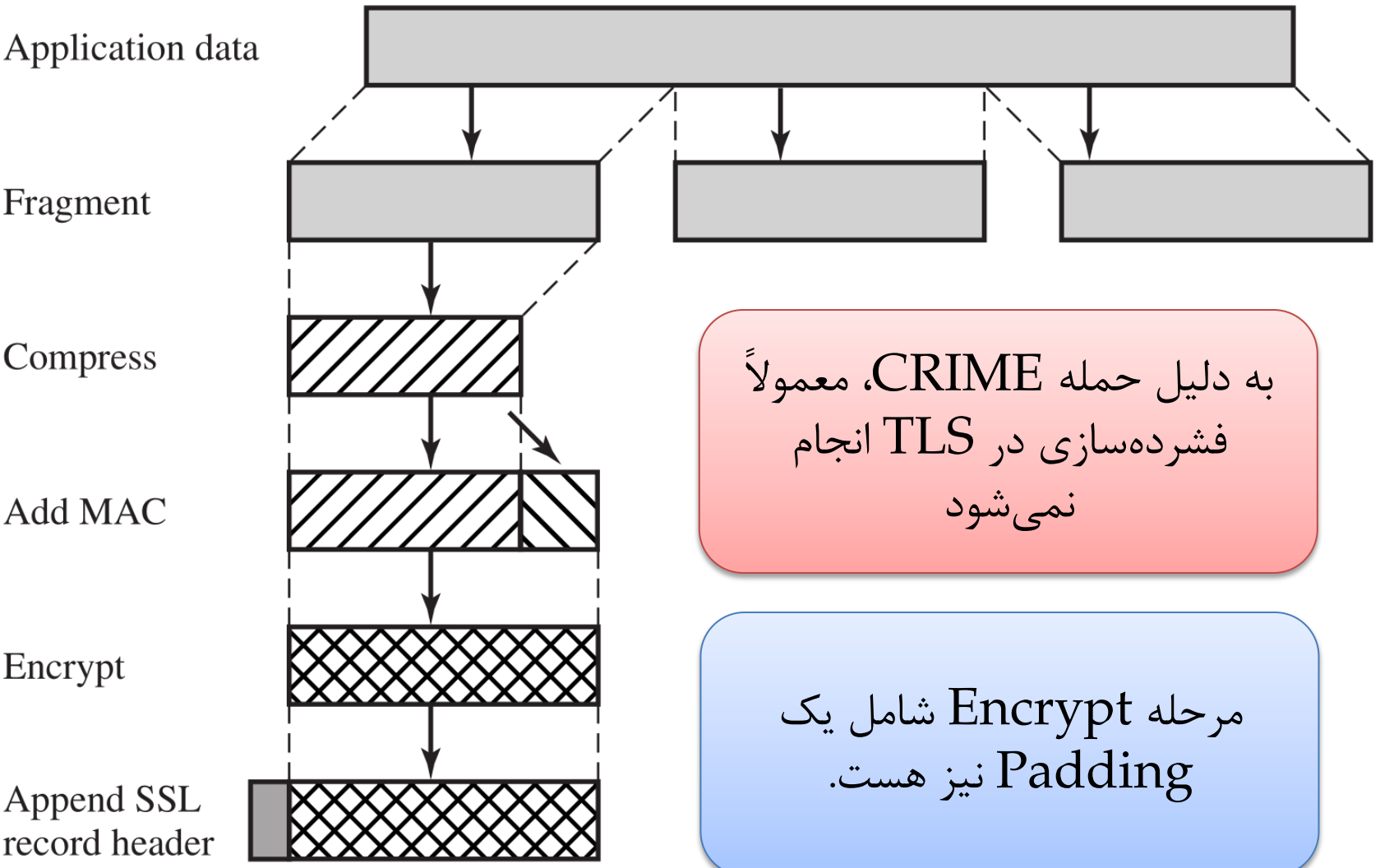
- زیر پروتکلهای دستداد، تغییر رمز، هشدار و لایه بالایی روی پروتکل رکورد اجرا می‌شوند؛ همان طور که HTTP روی TCP اجرا می‌شود.

☜ پروتکل رکورد سرآیندهای لازم را به آنها افزوده و در صورت لزوم رمزنگاری انجام می‌دهد.

اجرای پروتکلها روی زیر پروتکل رکورد



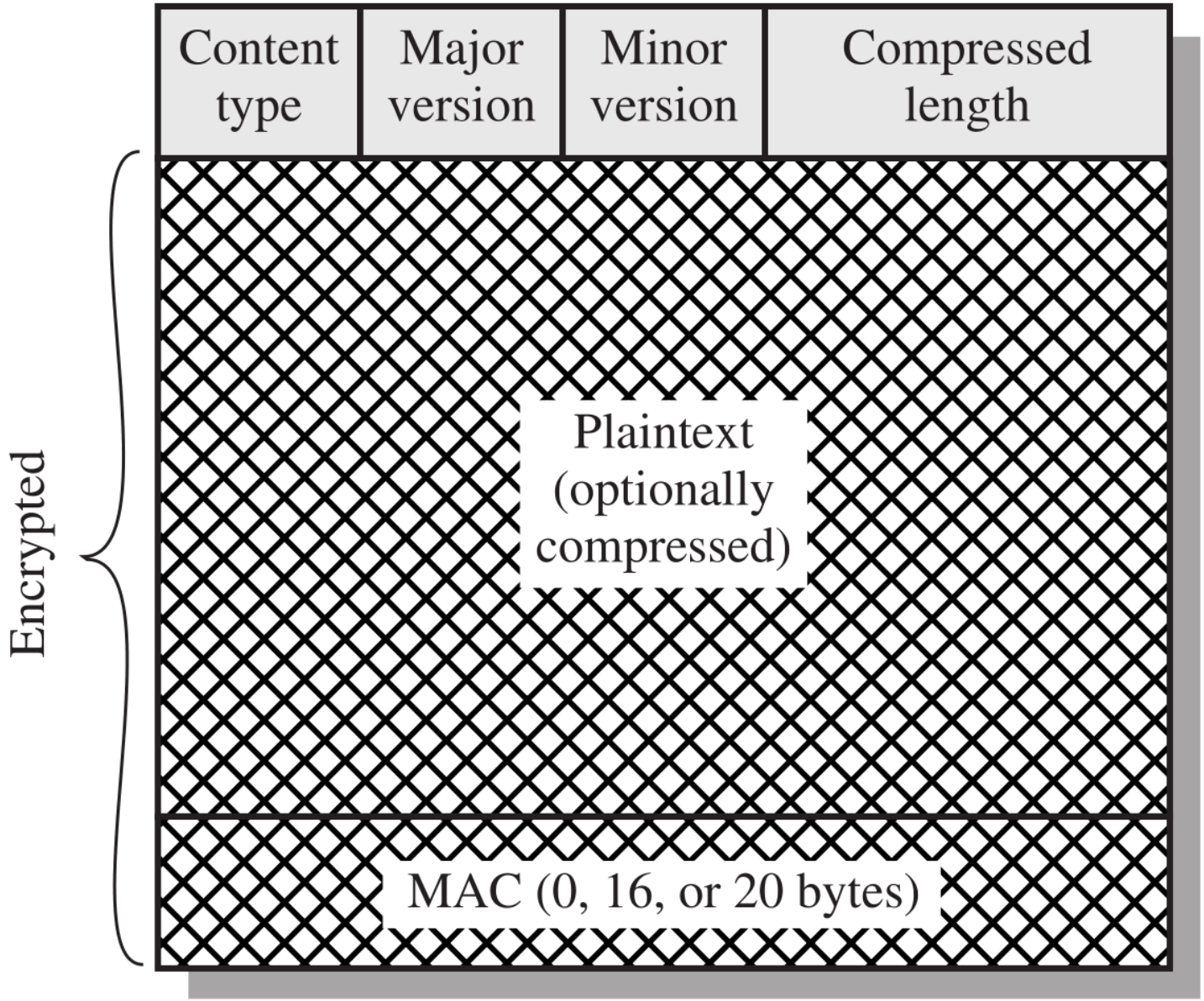
عملیات زیر پروتکل رکورد



به دلیل حمله CRIME، معمولاً فشرده‌سازی در TLS انجام نمی‌شود

مرحله Encrypt شامل یک Padding نیز هست.

قالب بسته‌های زیر پروتکل رکورد



زیر پروتکل تغییر رمز

- در خلال پروتکل دستداد، هریک از کارخواه و کارگزار پارامترهای امنیتی مورد نظر خود را می‌فرستند.
- پس از پایان کار، با ارسال یک زیر پروتکل «تغییر رمز»، پارامترها نهایی می‌شوند.

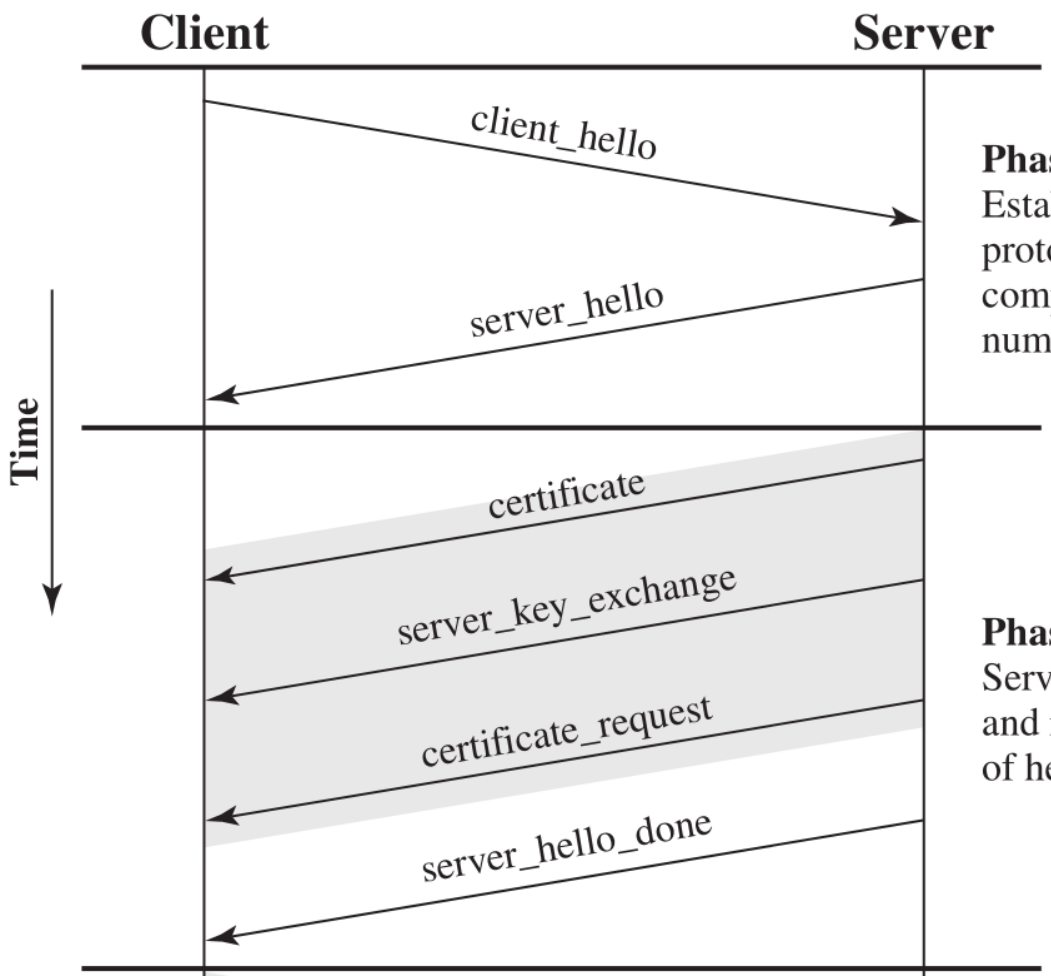
☜ با دریافت پیغام تغییر رمز، حالت معلق (pending) هر طرف به حالت جاری (current) تبدیل می‌شود.

- زیر پروتکل «تغییر رمز» کوچکترین پروتکل امنیت شبکه است.

☜ شامل فقط ۱ بایت با مقدار ۱!

چهار فاز تبادل پارامترهای امنیتی – ۱

پیامهای خاکستری اختیاری یا
وابسته به موقعیت هستند.

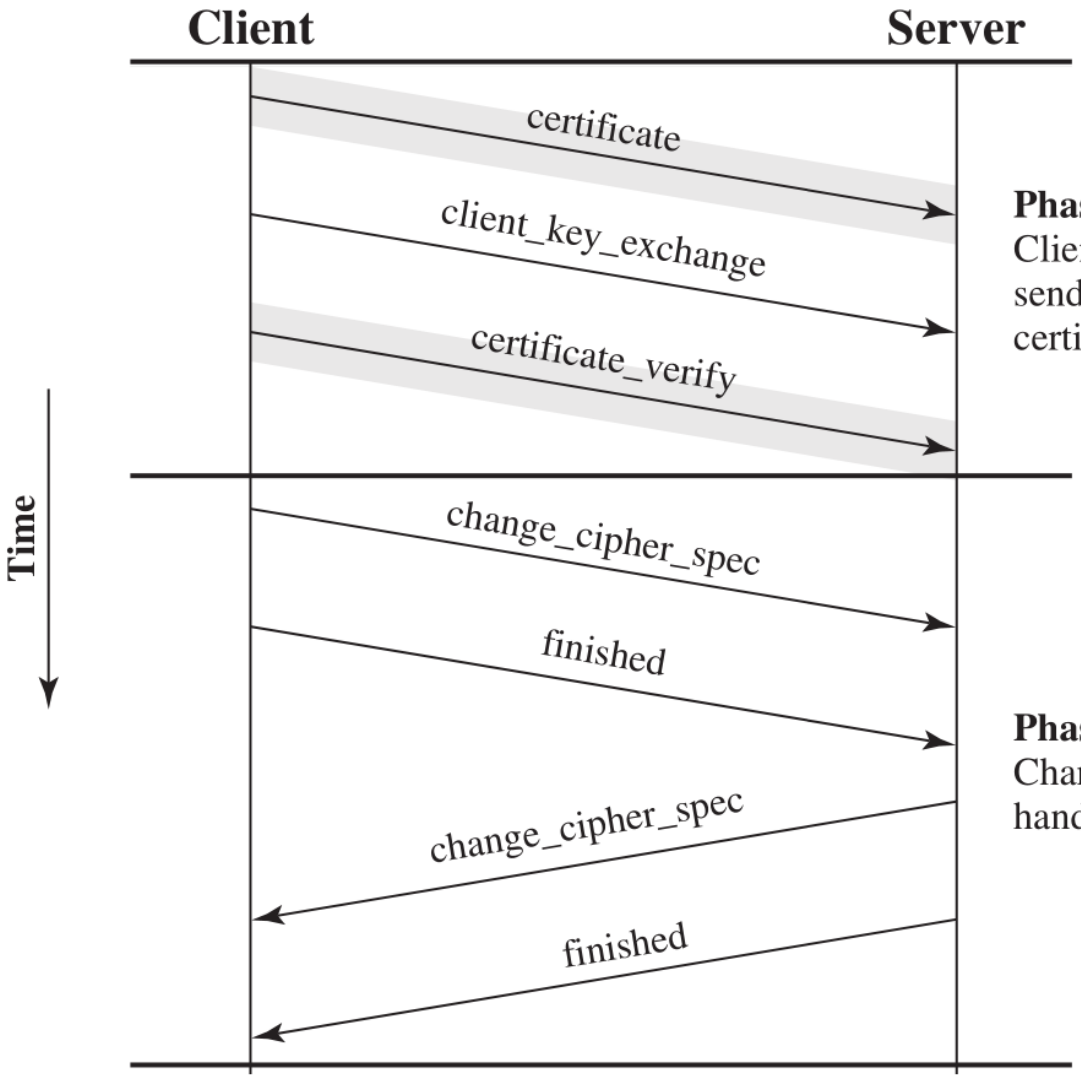


Phase 1
Establish security capabilities, including protocol version, session ID, cipher suite, compression method, and initial random numbers.

Phase 2
Server may send certificate, key exchange, and request certificate. Server signals end of hello message phase.

چهار فاز تبادل پارامترهای امنیتی – ۲

پیامهای خاکستری اختیاری یا وابسته به موقعیت هستند.



Phase 3
Client sends certificate if requested. Client sends key exchange. Client may send certificate verification.

Phase 4
Change cipher suite and finish handshake protocol.

علت وجود پیامهای اختیاری یا وابسته به موقعیت – ۱

- انواع مدلهای اعتماد:

☜ طرفین هیچ کلید مشترکی از هم ندارند (Anonymous DH)

☜ طرفین از هم کلید متقارن (Pre-Shared Key یا PSK) دارند؛ مشهور به TLS-PSK.

☜ کارگزار، کارخواه، یا هر دو از هم گواهی دیجیتال دارند.

- مبتنی بر RSA (دو نوع: RSA فقط برای امضا؛ RSA برای امضا و رمز)
- مبتنی بر DSA (دو نوع: شامل پارامترهای DH؛ بدون پارامترهای DH)

Ephemeral DH Fixed DH

علت وجود پیامهای اختیاری یا وابسته به موقعیت – ۲

- نوع پروتکل مورد استفاده جهت تبادل کلید:
  - **انتقال کلید (Transport):** معمولاً مبتنی بر RSA
  - **تبادل کلید:** معمولاً مبتنی بر DH
- استفاده از DH به دلیل فراهم آوردن امنیت پیشرو ترجیح دارد.
- DH با امنیت مساوی RSA شدیداً کندتر است.
  - استفاده از DH روی خمهای بیضوی (Elliptic Curves) معروف به ECDHE (E انتهایی کوته‌نوشت Exchange است).

Pre-Master Secret

- کلیدی که در پروتکل دستداد تبادل می‌شود، مقداری به نام Pre-Master Secret است.
- با استفاده از Pre-Master Secret، شش مقدار مخفی محاسبه می‌شود:
  - Client write MAC secret
  - Server write MAC secret
  - Client write encryption key
  - Server write encryption key
  - Client write encryption IV
  - Server write encryption IV

زیر پروتکل هشدار

- در صورتی که در حین اجرای پروتکل SSL/TLS خطایی رخ دهد، یا طرفین بخواهند پیامهای کنترلی بفرستند، زیر پروتکل هشدار اجرا می‌شود.

☜ **شامل دو بایت:** سطح هشدار، و کد هشدار

- سطح هشدار می‌تواند warning (مقدار ۱) یا fatal (مقدار ۲) باشد.

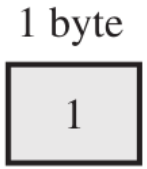
- سطح هشدار fatal بلافاصله باعث بسته شدن اتصال می‌شود.

☜ سایر اتصالها روی نشست جاری ممکن است ادامه یابند، ولی اتصال جدیدی اجازه تشکیل نخواهد داشت.

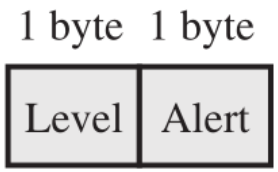
نمونه‌هایی از زیر پروتکل هشدار

- unexpected_message
- bad_record_mac
- handshake_failure
- certificate_revoked
- certificate_expired
- close_notify → نمونه‌ای از پیام کنترلی

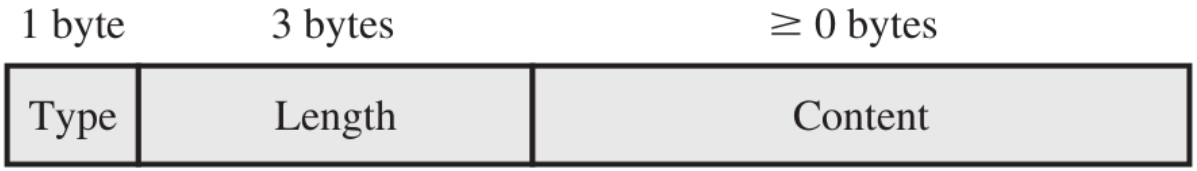
انواع Payload زیر پروتکل رکورد در یک نگاه



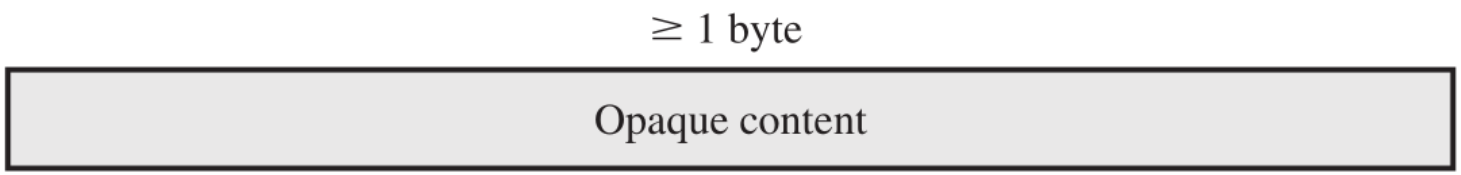
(a) Change Cipher Spec Protocol



(b) Alert Protocol



(c) Handshake Protocol

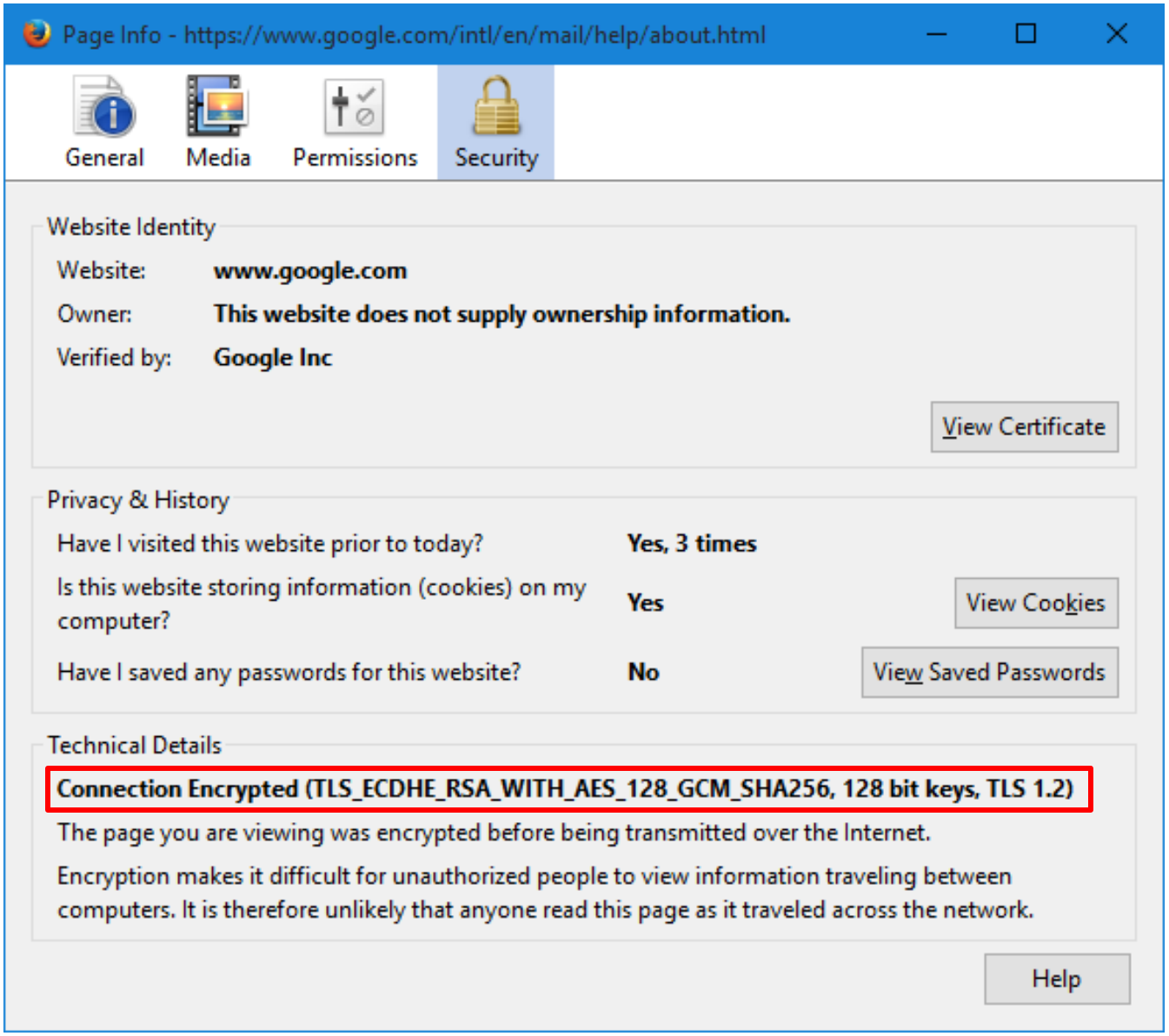


(d) Other Upper-Layer Protocol (e.g., HTTP)

فهرست مطالب

- معرفی و تاریخچه
- SSL/TLS در سطح بالا
- **TLS در عمل**
- جزئیات TLS
- Heartbleed

مثال: اطلاعات فایرفاکس از رمزنگاری Gmail



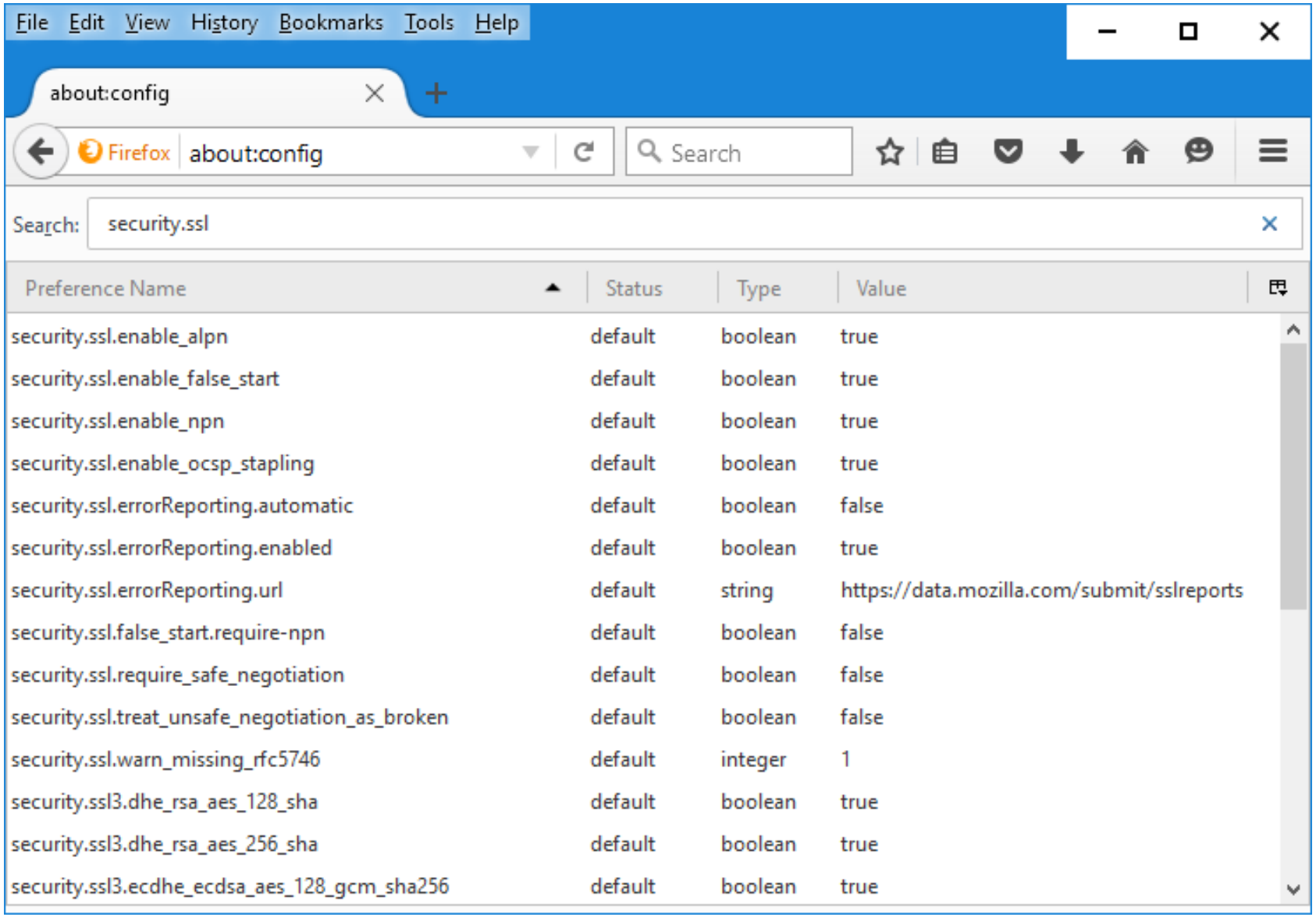
واکاوی بیشتر

- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256, 128 bit keys, TLS 1.2
  - **ECDHE:** Elliptic Curve DH Exchange
  - **RSA:** Gmail Public Key Type
  - **AES 128:** Symmetric Key Cipher
  - **GCM:** Mode of Encryption
  - **SHA256:** Hash Algorithm (for MAC)
  - **TLS 1.2:** TLS Version

نقشه راه

- در ادامه، سعی می‌کنیم تا TLS را در عمل بررسی کنیم.
- با توجه به گستردگی پروتکل، امکان بررسی تمام حالتها وجود ندارد.
- دو حالت خاص:
  - **حالت ۱:** کلید عمومی RSA؛ انتقال کلید با RSA
  - **حالت ۲:** کلید عمومی RSA؛ تبادل کلید DH
- Apache + OpenSSL روی Ubuntu به عنوان کارگزار
- Firefox به عنوان کارخواه

جزئیات پیکربندی TLS با about:config در فایرفاکس



Preference Name	Status	Type	Value
security.ssl.enable_alpn	default	boolean	true
security.ssl.enable_false_start	default	boolean	true
security.ssl.enable_npn	default	boolean	true
security.ssl.enable_ocsp_stapling	default	boolean	true
security.ssl.errorReporting.automatic	default	boolean	false
security.ssl.errorReporting.enabled	default	boolean	true
security.ssl.errorReporting.url	default	string	https://data.mozilla.com/submit/sslreports
security.ssl.false_start.require-npn	default	boolean	false
security.ssl.require_safe_negotiation	default	boolean	false
security.ssl.treat_unsafe_negotiation_as_broken	default	boolean	false
security.ssl.warn_missing_rfc5746	default	integer	1
security.ssl3.dhe_rsa_aes_128_sha	default	boolean	true
security.ssl3.dhe_rsa_aes_256_sha	default	boolean	true
security.ssl3.ecdhe_ecdsa_aes_128_gcm_sha256	default	boolean	true

تنظیمات Apache + OpenSSL

- تولید گواهی دیجیتال کارگزار توسط OpenSSL:

```
sudo mkdir /etc/apache2/ssl

sudo openssl req -x509 -nodes -days 1095 -newkey
rsa:2048 -out /etc/apache2/ssl/server.crt -keyout
/etc/apache2/ssl/server.key
```

- تنظیم پورت در `/etc/apache2/ports.conf`:

```
Listen 443
```

- فعال‌سازی و پیکربندی ماژول SSL آپاچی (mod_ssl)

پیکربندی ماژول mod_ssl

- فعال‌سازی:

```
sudo a2enmod ssl
```

- پیکربندی:

```
/etc/apache2/sites-available/default-ssl.conf
```

```
SSLCertificateFile    /etc/apache2/ssl/server.crt
SSLCertificateKeyFile /etc/apache2/ssl/server.key
```

- راه‌اندازی مجدد آپاچی:

```
sudo /etc/init.d/apache2 restart
```

تنظیمات امنیتی در ماژول mod_ssl

- آنچه تا کنون گفته شد، حداقل تنظیمات برای راه‌اندازی HTTPS در آپاچی بود.
- برای امنیت بیشتر، باید تنظیمات دیگری در فایل **ssl.conf** آپاچی انجام داد (پس از انجام تنظیمات آپاچی باید مجدداً راه‌اندازی شود).
- در اسلایدهای بعدی دو تنظیم مهم را بررسی می‌کنیم:
  - ☜ نسخه SSL/TLS
  - ☜ الگوریتمهای رمز و پروتکلهای مورد استفاده

نسخه SSL/TLS

```
# The protocols to enable.
# Available values:
#     all, SSLv3, TLSv1, TLSv1.1, TLSv1.2
# SSL v2  is no longer supported
```

SSLProtocol all

- در حال حاضر امن‌ترین تنظیم، `TLSv1.2` است.

☜ البته لازم است پشتیبانی کارخواه‌ها از این نسخه از پروتکل در نظر گرفته شود.

الگوریتمهای رمز و پروتکلهای مورد استفاده

```
# SSL Cipher Suite:
# List the ciphers that the client is
# permitted to negotiate. See the
# ciphers(1) man page from the openssl
# package for list of all available options.
# Enable only secure ciphers:

SSLCipherSuite HIGH:MEDIUM:!aNULL:!MD5
```

- علامت تعجب (!) به معنی عدم استفاده است.

☜ aNull یعنی پروتکل بدون authentication (فعلاً معادل Anonymous DH).

- در مثال فوق فقط رمزهای با امنیت متوسط و بالا مورد قبولند.

مجبور کردن کارگزار به عدم استفاده از DH

- به طور پیش‌فرض، کارگزار از پروتکل ECDH برای تبادل کلید استفاده می‌کند.

☜ در جهت حفظ محرمانگی پیشرو

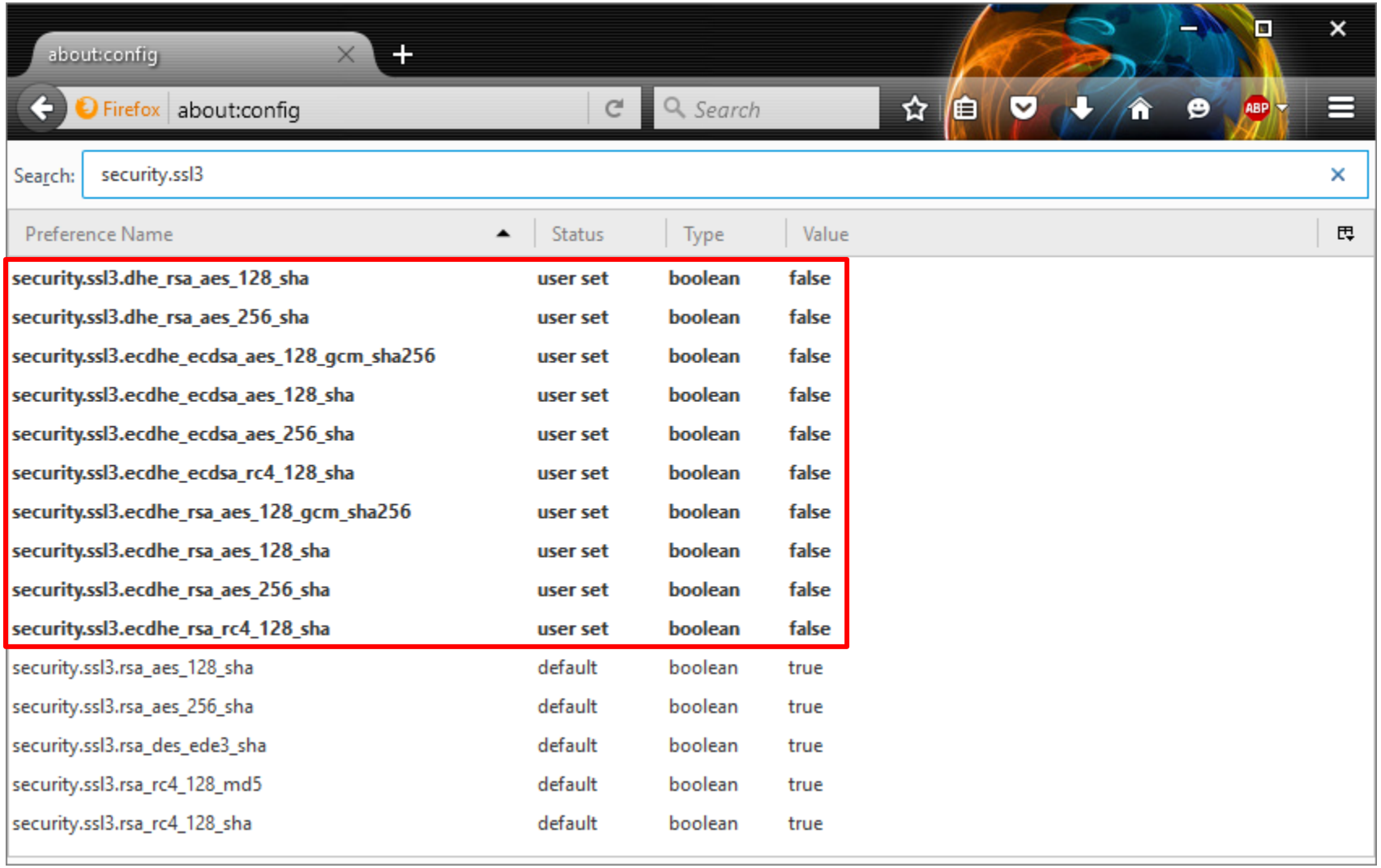
- در حالت اول می‌خواهیم انتقال کلید از طریق RSA انجام شود؛ به همین دلیل ECDH را سمت کارگزار غیر فعال می‌کنیم.

/etc/apache2/mods-enabled/ssl.conf

```
SSLCipherSuite !ECDH:!DH:HIGH:MEDIUM:!aNULL:!MD5
```

مجبور کردن مرورگر به عدم استفاده از DH

- همچنین می‌توانستیم از فایرفاکس بخواهیم که از DH استفاده نکند.



حالت ۱ – تعامل RSA بین کارخواه و کارگزار

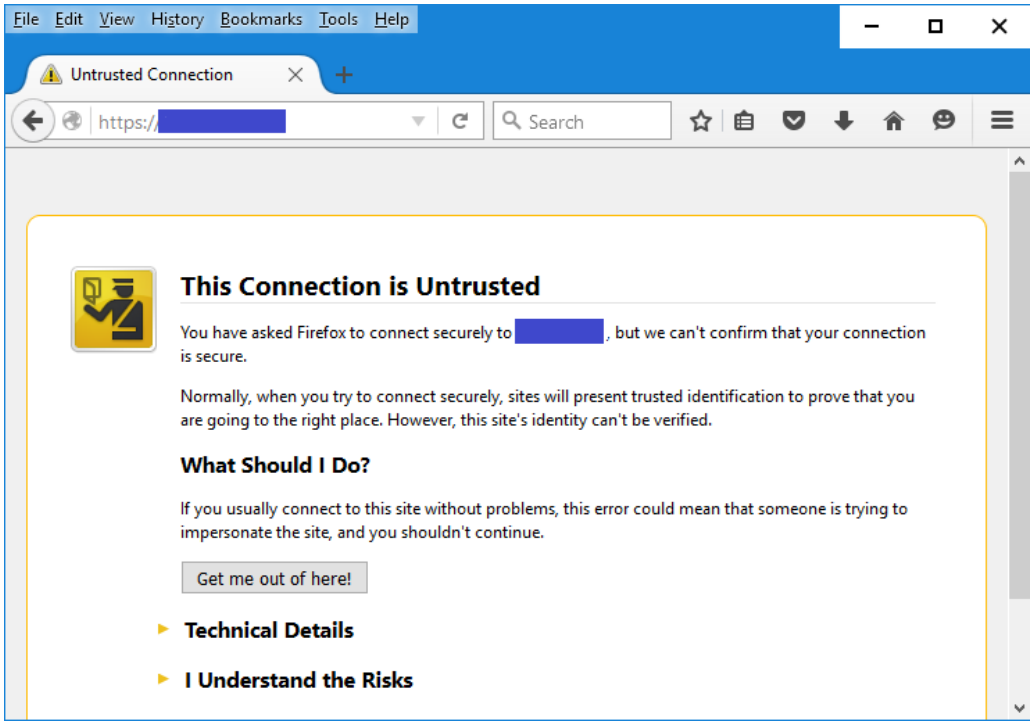
- با استفاده از Wireshark تعامل را شنود می‌کنیم.

Alert.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.00	client	server	TLSv1.2	255	Client Hello
2	0.80	server	client	TLSv1.2	1288	Server Hello, Certificate, Server Key Exchange, Server Hello Done
3	0.80	client	server	TLSv1.2	220	Client Key Exchange, Change Cipher Spec, Hello Request, Hello Request
4	0.82	client	server	TLSv1.2	121	Encrypted Alert
5	1.60	server	client	TLSv1.2	352	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message



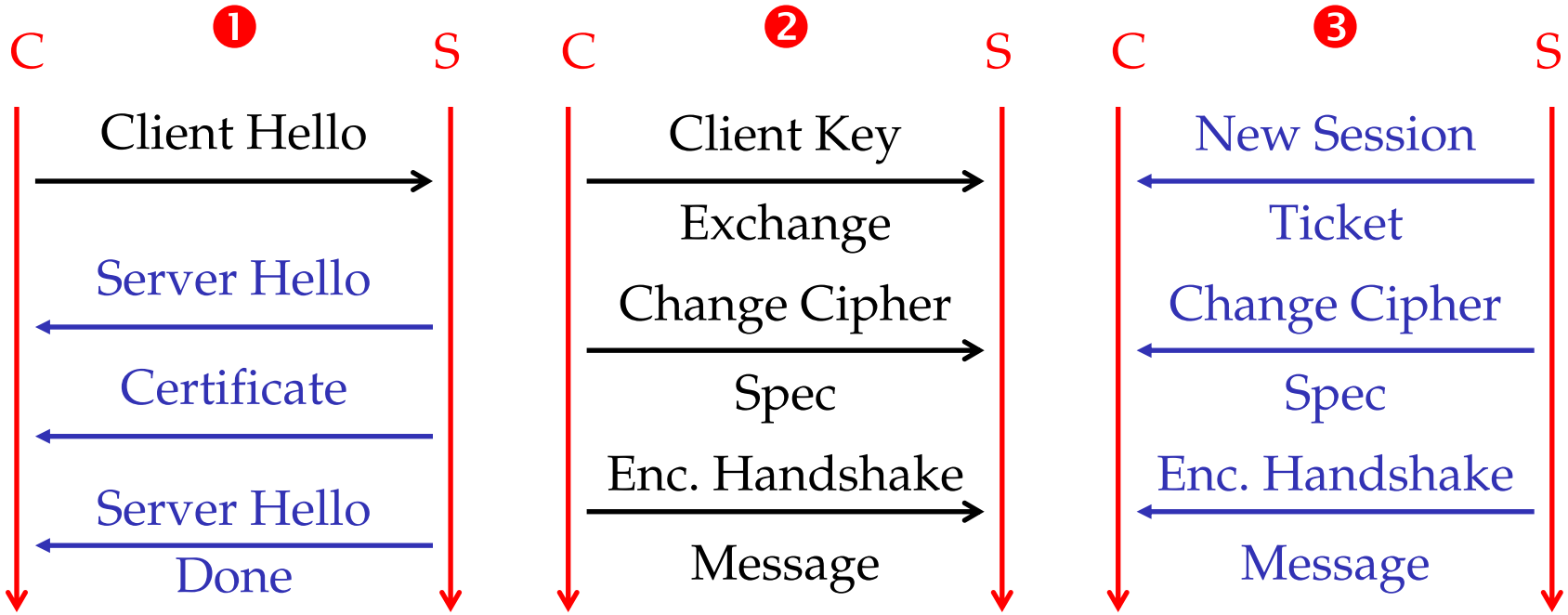
در صورت اعتماد به کلید عمومی کارگزار و ادامه پروتکل

SSL-RSA.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

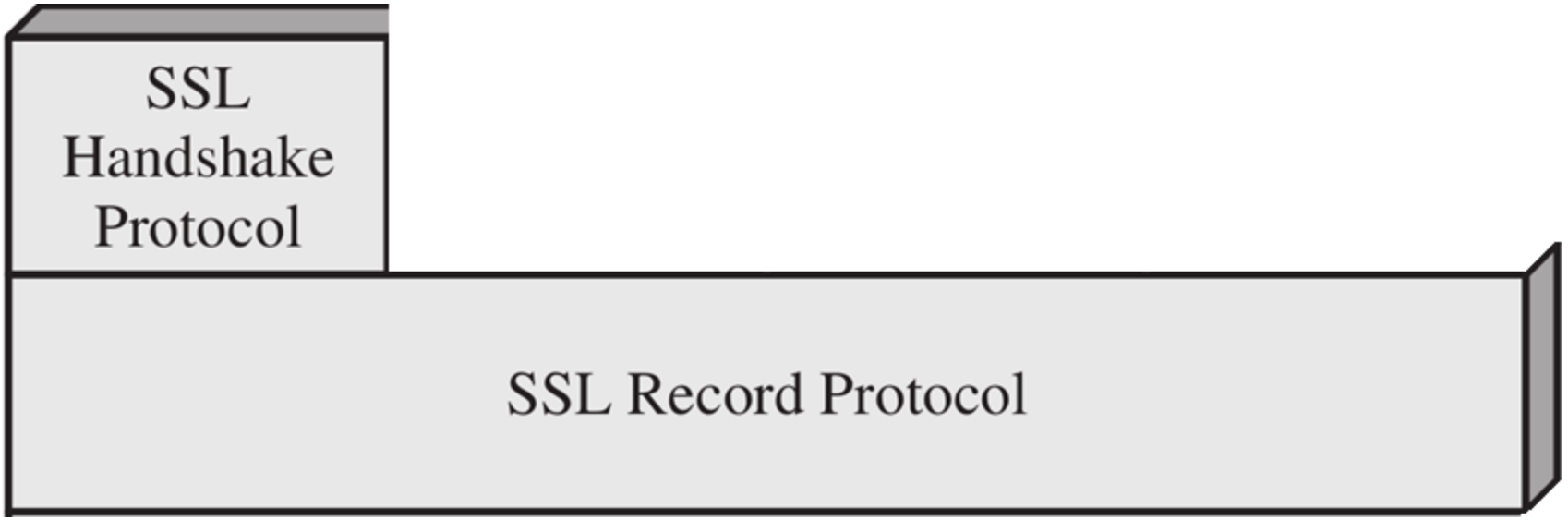
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000	client	server	TLSv1.2	182	Client Hello
2	0.429	server	client	TLSv1.2	902	Server Hello, Certificate, Server Hello Done
3	0.430	client	server	TLSv1.2	396	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
4	0.863	server	client	TLSv1.2	336	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
5	0.864	client	server	TLSv1.2	395	Application Data
6	1.296	server	client	TLSv1.2	853	Application Data, Application Data, Application Data
7	1.367	client	server	TLSv1.2	395	Application Data
8	1.799	server	client	TLSv1.2	656	Application Data, Application Data
9	1.800	client	server	TLSv1.2	395	Application Data
10	2.228	server	client	TLSv1.2	656	Application Data, Application Data



Client Hello روی پروتکل رکورد

```
> Frame 1: 182 bytes on wire (1456 bits), 182 bytes captured (1456 bits) on interface 0
> Ethernet II, Src: IntelCor_d9:62:f5 (ac:72:89:d9:62:f5), Dst: D-LinkIn_d3:db:4c (70:62:b8:d3:db:4c)
> Internet Protocol Version 4, Src: client (192.168.1.4), Dst: server (           )
> Transmission Control Protocol, Src Port: 6551 (6551), Dst Port: 443 (443), Seq: 1, Ack: 1, Len: 128
v Secure Sockets Layer
  v TLSv1.2 Record Layer: Handshake Protocol: Client Hello
      Content Type: Handshake (22)
      Version: TLS 1.0 (0x0301)
      Length: 123
    > Handshake Protocol: Client Hello
```

پروتکل رکورد از نسخه 1.0 از TLS استفاده کرده است.



داخل پیام Client Hello

```
▾ Handshake Protocol: Client Hello
    Handshake Type: Client Hello (1)
    Length: 119
    Version: TLS 1.2 (0x0303)
  ▾ Random
      GMT Unix Time: Oct 11, 2105 18:06:07.000000000 Iran Standard Time
      Random Bytes: 66d6ef331b0b9071cdec232cc5ab501c9cabce9406e6ffb4...
    Session ID Length: 0
    Cipher Suites Length: 6
  ▾ Cipher Suites (3 suites)
      Cipher Suite: TLS_RSA_WITH_AES_128_CBC_SHA (0x002f)
      Cipher Suite: TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
      Cipher Suite: TLS_RSA_WITH_3DES_EDE_CBC_SHA (0x000a)
    Compression Methods Length: 1
  › Compression Methods (1 method)
    Extensions Length: 72
  › Extension: renegotiation_info
  › Extension: SessionTicket TLS
  › Extension: next_protocol_negotiation
  › Extension: Application Layer Protocol Negotiation
  › Extension: status_request
  › Extension: signature_algorithms
```

پروتکل دستداد از نسخه 1.2 از TLS استفاده کرده است.

با محدود کردن رمزها در فایرفاکس، هیچ روشی غیر از RSA برای تبادل کلید پیشنهاد نمی‌شود.

پاسخ کارگزار

```
Secure Sockets Layer
  TLSv1.2 Record Layer: Handshake Protocol: Server Hello
  TLSv1.2 Record Layer: Handshake Protocol: Certificate
  TLSv1.2 Record Layer: Handshake Protocol: Server Hello Done
```

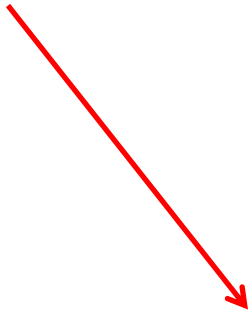
داخل پیامهای رکورد

```
Secure Sockets Layer
  TLSv1.2 Record Layer: Handshake Protocol: Server Hello
    Content Type: Handshake (22)
    Version: TLS 1.2 (0x0303)
    Length: 53
    Handshake Protocol: Server Hello
  TLSv1.2 Record Layer: Handshake Protocol: Certificate
    Content Type: Handshake (22)
    Version: TLS 1.2 (0x0303)
    Length: 776
    Handshake Protocol: Certificate
  TLSv1.2 Record Layer: Handshake Protocol: Server Hello Done
    Content Type: Handshake (22)
    Version: TLS 1.2 (0x0303)
    Length: 4
    Handshake Protocol: Server Hello Done
```

دقت: کارگزار نسخه 1.2 از TLS را پیشنهاد می‌دهد.

داخل پیام Server Hello

```
▾ Handshake Protocol: Server Hello
    Handshake Type: Server Hello (2)
    Length: 49
    Version: TLS 1.2 (0x0303)
  ▾ Random
      GMT Unix Time: Oct  2, 2043 23:47:27.000000000 Iran Standard Time
      Random Bytes: 3338f1835d4e202a847a51f89e6017c8de2102b0091362c4...
    Session ID Length: 0
    Cipher Suite: TLS_RSA_WITH_AES_128_CBC_SHA (0x002f)
    Compression Method: null (0)
    Extensions Length: 9
  › Extension: renegotiation_info
  › Extension: SessionTicket TLS
```



کـارگزار یکـی از رمزهـای پیشـنهاد شـده
توسط مرورگر را می‌پذیرد.

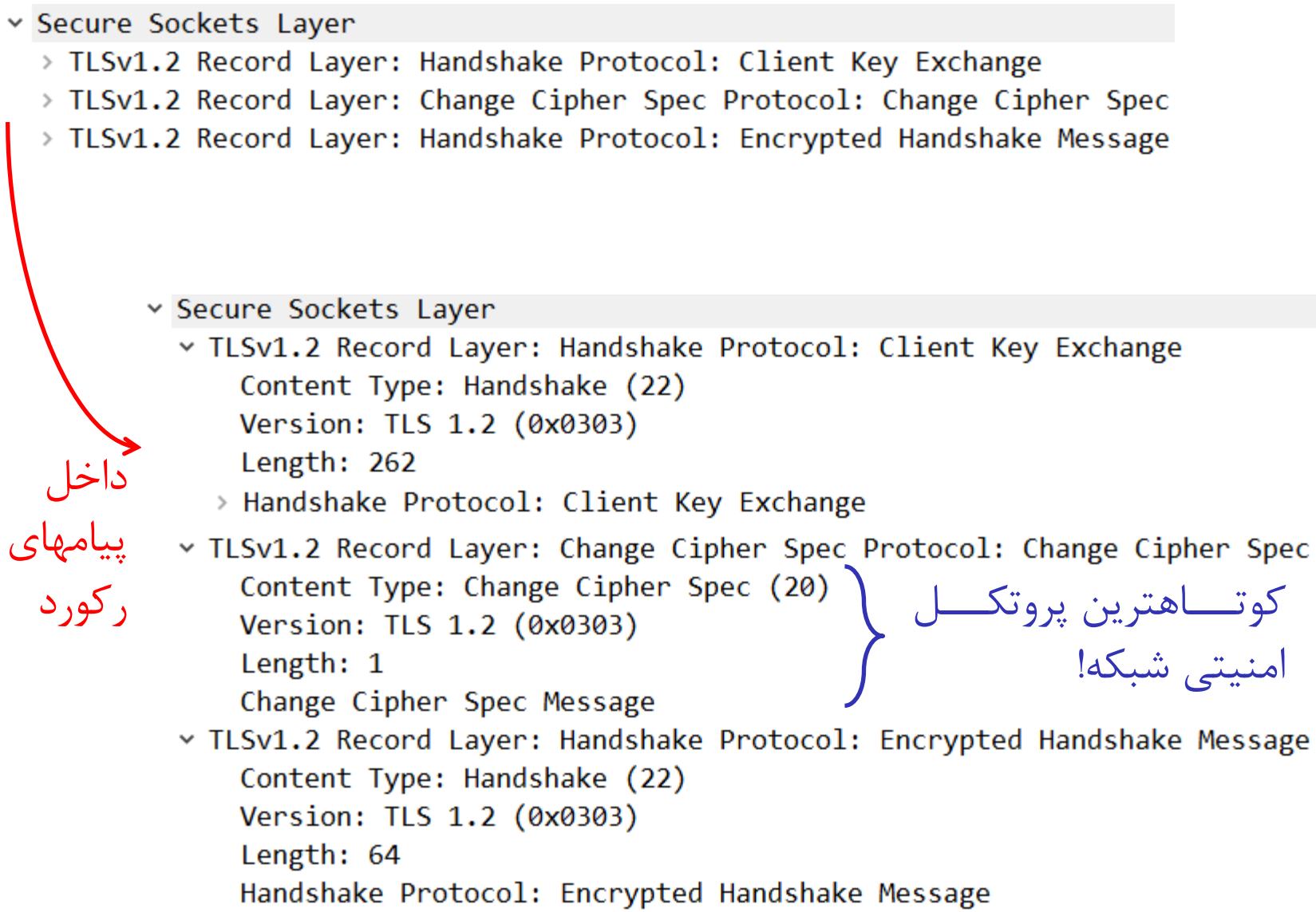
داخل پیام Certificate

```
▾ Handshake Protocol: Certificate
    Handshake Type: Certificate (11)
    Length: 772
    Certificates Length: 769
  ▾ Certificates (769 bytes)
      Certificate Length: 766
    ▾ Certificate: 308202fa308201e2a003020102020900b6baf75031e60163...
      ▾ signedCertificate
          version: v3 (2)
          serialNumber: -5279635689331883677
        › signature (sha256WithRSAEncryption)
        › issuer: rdnSequence (0)
        › validity
        › subject: rdnSequence (0)
        › subjectPublicKeyInfo
        › extensions: 1 item
      ▾ algorithmIdentifier (sha256WithRSAEncryption)
          Algorithm Id: 1.2.840.113549.1.1.11 (sha256WithRSAEncryption)
        Padding: 0
        encrypted: 4ce947b4b16027490a24d42b881636f4ba3c7c25a2880136...
```

داخل پیام Server Hello Done

```
▾ Handshake Protocol: Server Hello Done
    Handshake Type: Server Hello Done (14)
    Length: 0
```

پاسخ کارخواه



داخل پیام Client Key Exchange

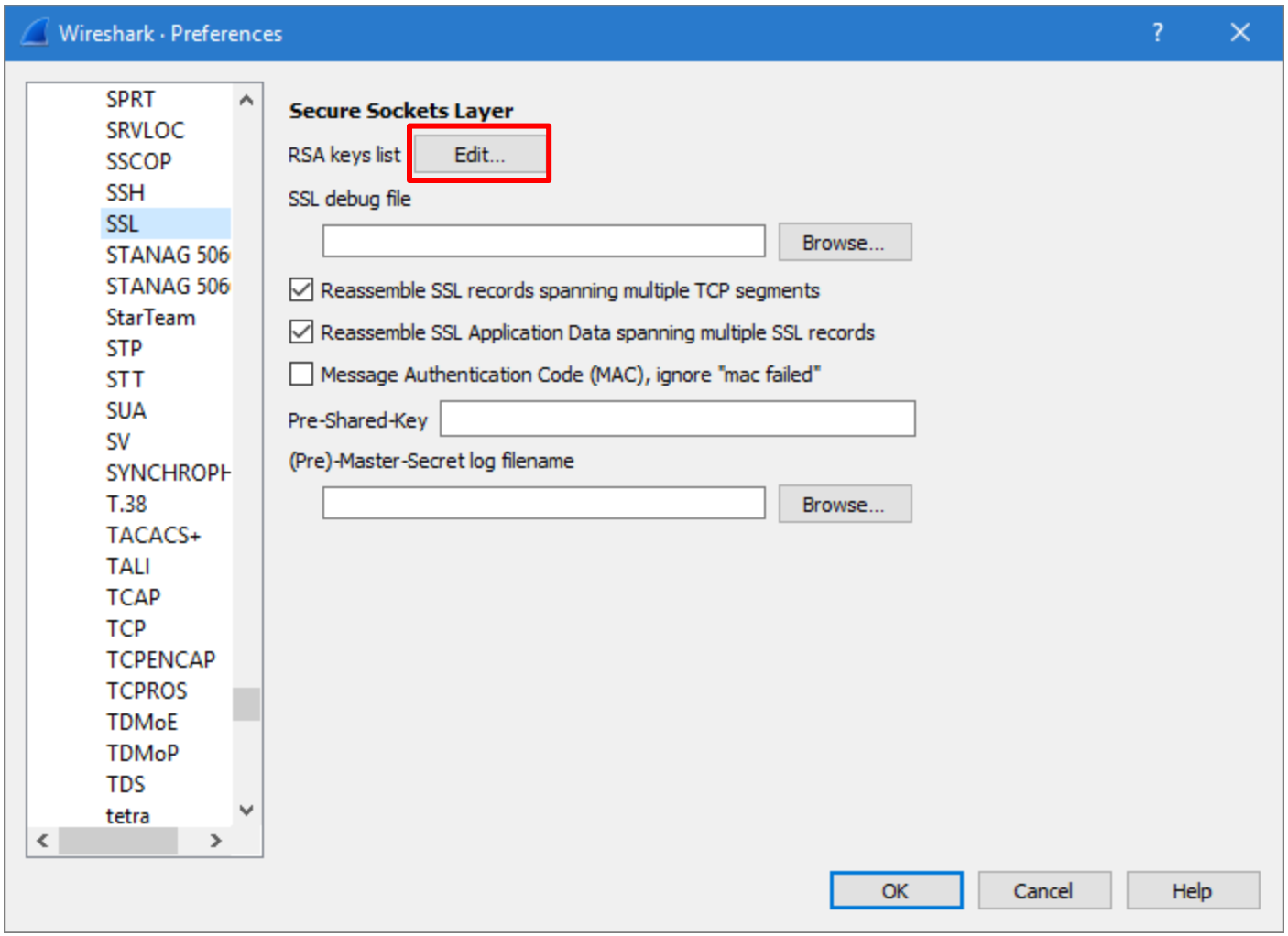
```
▾ Handshake Protocol: Client Key Exchange
    Handshake Type: Client Key Exchange (16)
    Length: 258
  ▾ RSA Encrypted PreMaster Secret
      Encrypted PreMaster length: 256
      Encrypted PreMaster: 58e0c7eb53c31610fc60645ca869b378b9cfca0223b28c02...
```

- کارخواه، Pre-Master Secret را تولید نموده، آن را با کلید عمومی کارگزار رمز و ارسال می‌کند.

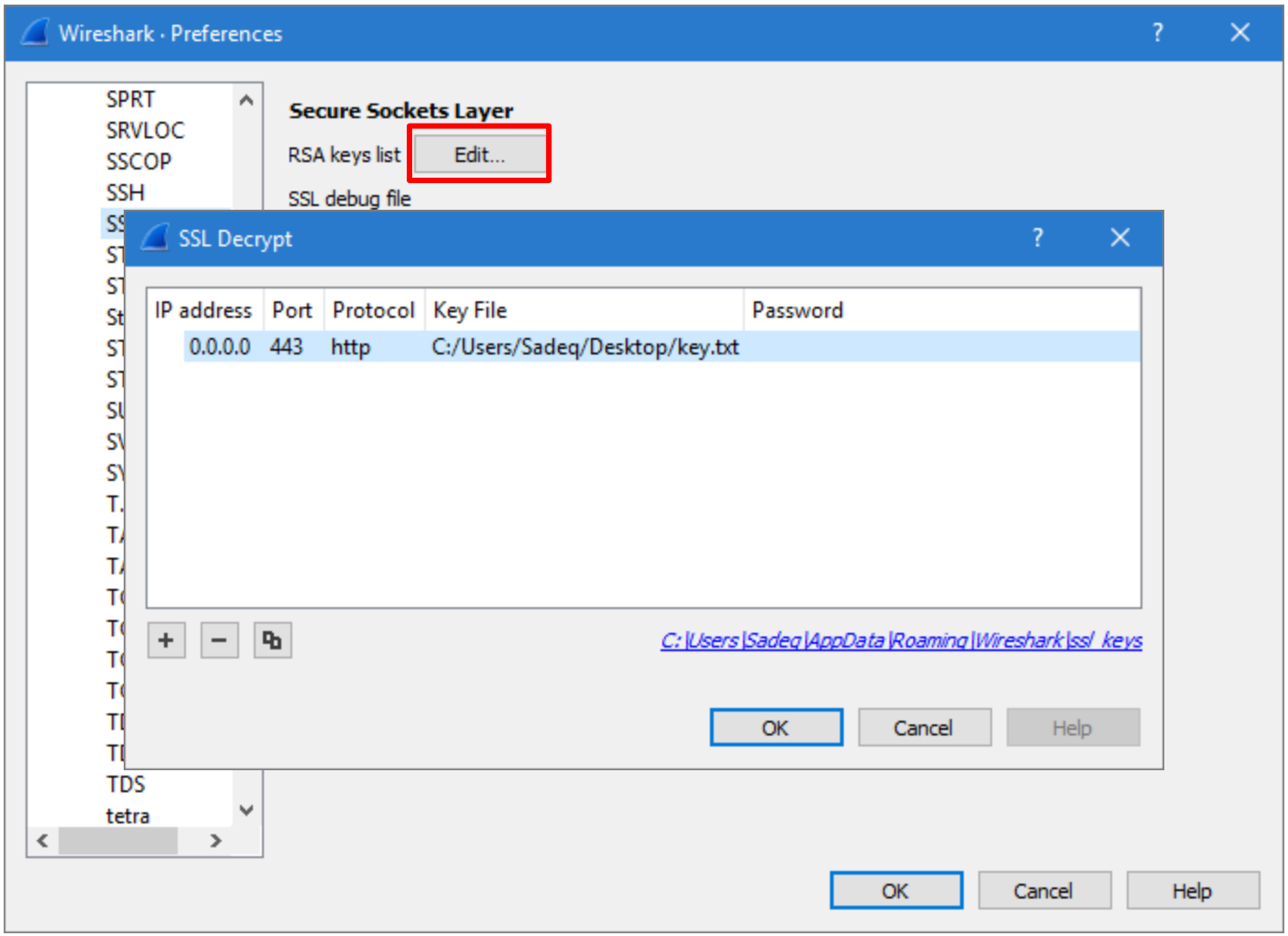
☜ ۶ کلید نشست از روی Pre-Master Secret استخراج شده و از این پس تقریباً همه چیز رمز شده خواهد بود.

- **سؤال:** آیا می‌توان از Wireshark خواست که پیامها را رمزگشایی کند؟

رمزگشایی پیامها توسط Wireshark با کلید خصوصی کارگزار



رمزگشایی پیامها توسط Wireshark با کلید خصوصی کارگزار



کل پروتکل – قبل و بعد از رمزگشایی

SSL-RSA.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000	client	server	TLSv1.2	182	Client Hello
2	0.429	server	client	TLSv1.2	902	Server Hello, Certificate, Server Hello Done
3	0.430	client	server	TLSv1.2	396	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
4	0.863	server	client	TLSv1.2	336	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
5	0.864	client	server	TLSv1.2	395	Application Data
6	1.296	server	client	TLSv1.2	853	Application Data, Application Data, Application Data
7	1.367	client	server	TLSv1.2	395	Application Data
8	1.799	server	client	TLSv1.2	656	Application Data, Application Data
9	1.800	client	server	TLSv1.2	395	Application Data
10	2.228	server	client	TLSv1.2	656	Application Data, Application Data

SSL-RSA.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

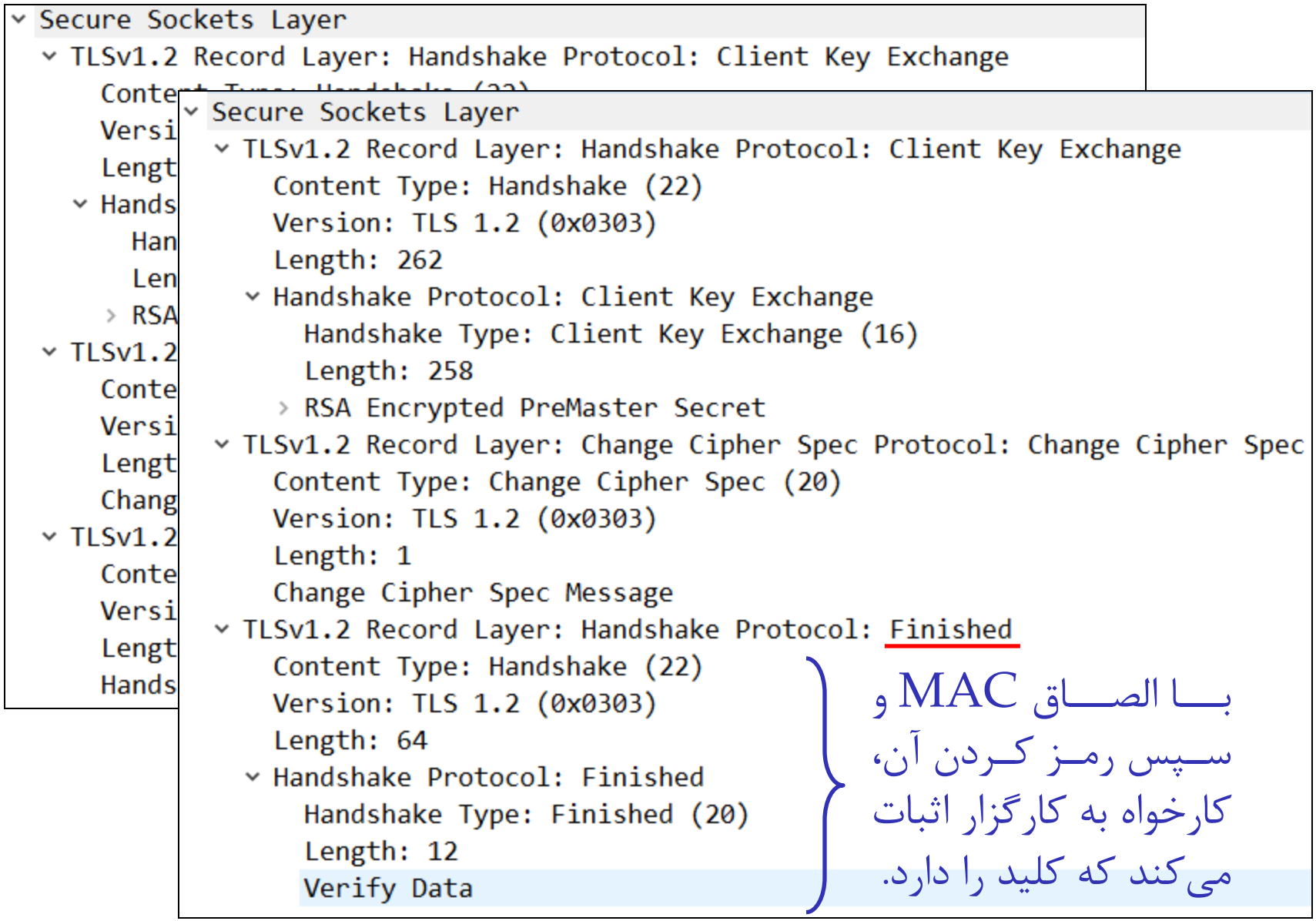
Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
1	0…	client	server	TLSv1.2	182	Client Hello
2	0…	server	client	TLSv1.2	902	Server Hello, Certificate, Server Hello Done
3	0…	client	server	TLSv1.2	396	Client Key Exchange, Change Cipher Spec, Finished
4	0…	server	client	TLSv1.2	336	New Session Ticket, Change Cipher Spec, Finished
5	0…	client	server	HTTP	395	GET / HTTP/1.1
6	1…	server	client	HTTP	853	HTTP/1.1 200 OK (text/html)HTTP/1.1 200 OK (text/html)
7	1…	client	server	HTTP	395	GET /favicon.ico HTTP/1.1
8	1…	server	client	HTTP	656	HTTP/1.1 404 Not Found (text/html)
9	1…	client	server	HTTP	395	GET /favicon.ico HTTP/1.1
10	2…	server	client	HTTP	656	HTTP/1.1 404 Not Found (text/html)

پاسخ کارخواه – قبل و بعد از رمزگشایی

```
Secure Sockets Layer
  TLSv1.2 Record Layer: Handshake Protocol: Client Key Exchange
    Content Type: Handshake (22)
    Version: TLS 1.2 (0x0303)
    Length: 262
    Handshake Protocol: Client Key Exchange
      Handshake Type: Client Key Exchange (16)
      Length: 258
      RSA Encrypted PreMaster Secret
  TLSv1.2 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
    Content Type: Change Cipher Spec (20)
    Version: TLS 1.2 (0x0303)
    Length: 1
    Change Cipher Spec Message
  TLSv1.2 Record Layer: Handshake Protocol: Encrypted Handshake Message
    Content Type: Handshake (22)
    Version: TLS 1.2 (0x0303)
    Length: 64
    Handshake Protocol: Encrypted Handshake Message
```

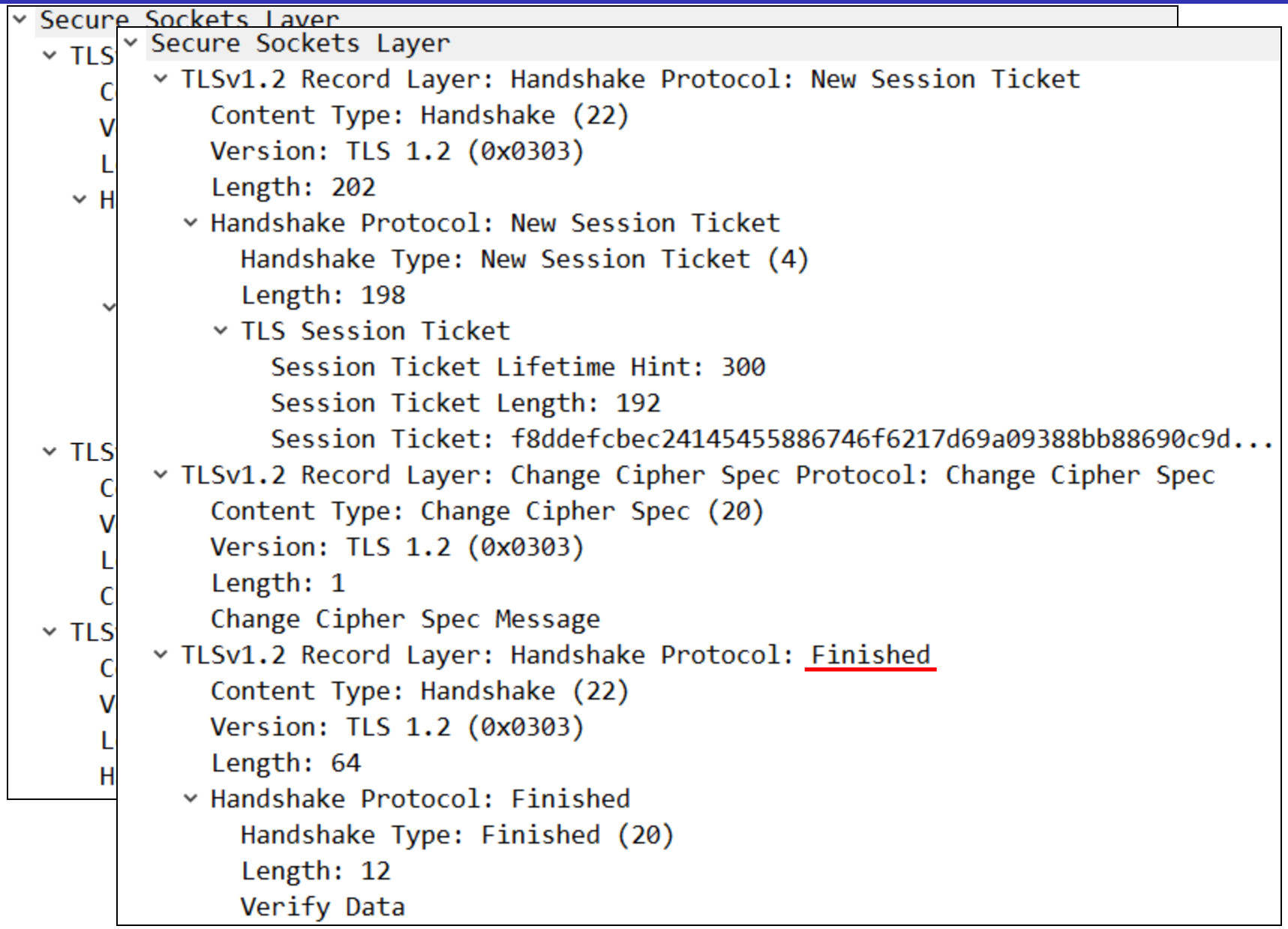
پاسخ کارخواه – قبل و بعد از رمزگشایی



پاسخ کارگزار – قبل و بعد از رمزگشایی

```
Secure Sockets Layer
  TLSv1.2 Record Layer: Handshake Protocol: New Session Ticket
    Content Type: Handshake (22)
    Version: TLS 1.2 (0x0303)
    Length: 202
    Handshake Protocol: New Session Ticket
      Handshake Type: New Session Ticket (4)
      Length: 198
      TLS Session Ticket
        Session Ticket Lifetime Hint: 300
        Session Ticket Length: 192
        Session Ticket: f8ddefcbec24145455886746f6217d69a09388bb88690c9d...
  TLSv1.2 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
    Content Type: Change Cipher Spec (20)
    Version: TLS 1.2 (0x0303)
    Length: 1
    Change Cipher Spec Message
  TLSv1.2 Record Layer: Handshake Protocol: Encrypted Handshake Message
    Content Type: Handshake (22)
    Version: TLS 1.2 (0x0303)
    Length: 64
    Handshake Protocol: Encrypted Handshake Message
```

پاسخ کارگزار – قبل و بعد از رمزگشایی



Session Ticket چیست؟

- Session Ticket یکی از توسعه‌های TLS است.
- در RFC 5077 تعریف شده است.
- هدف این است که نیازی نباشد کارگزار به ازای هر کارخواه، اطلاعات وضعیت (State) نگه دارد.
- اطلاعات وضعیت کارگزار از طریق Session Ticket در اختیار کارخواه قرار می‌گیرد.

☜ کارخواه در تماس‌های بعدی از آن استفاده می‌کند.

- از نظر مفهومی شبیه بلیت در کربروس است.

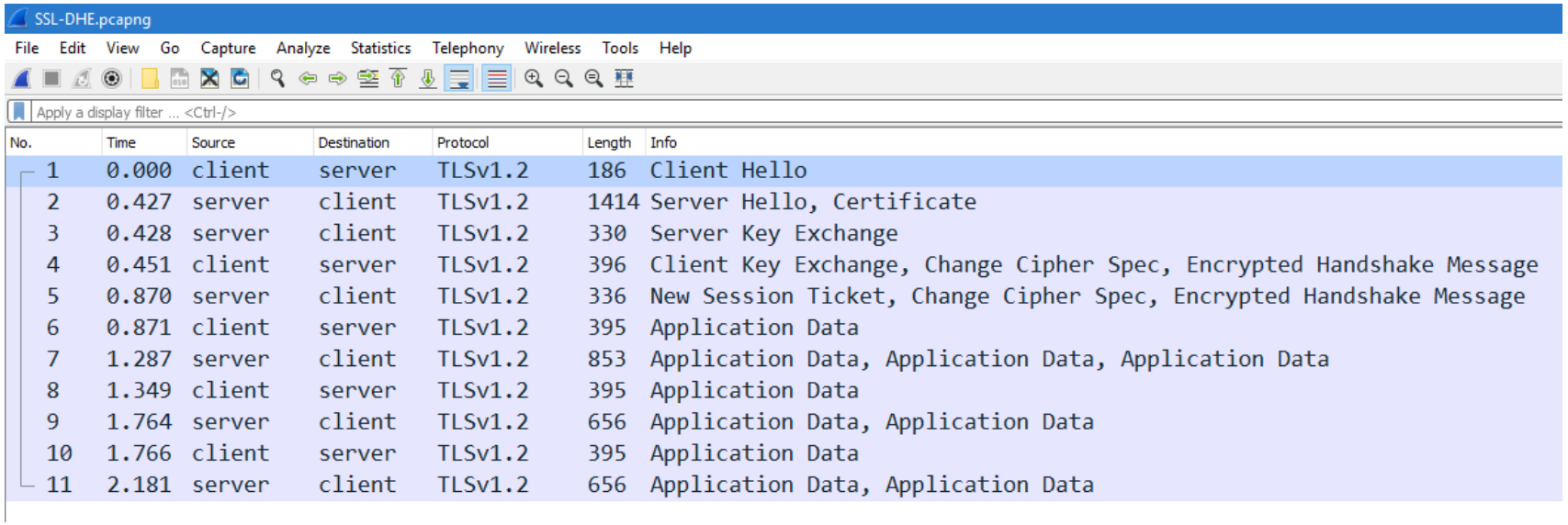
حالت ۲ – تعامل DH بین کارخواه و کارگزار

- در سمت کارگزار، تنظیم `DH ! : ECDH !` را حذف و آپاچی را مجدداً راه‌اندازی می‌کنیم.
- در سمت کارخواه، یکی از پروتکلهای مربوط به DH را فعال می‌نماییم (در بخش `about:config` فایرفاکس).

  ☜ برای سادگی، ECDH را فعال نمی‌کنیم.
- مجدداً تعامل را با Wireshark بررسی می‌کنیم.
- چرا حتی با داشتن کلید خصوصی کارگزار، Wireshark نمی‌تواند پیامهای رمز شده را رمزگشایی نماید؟

  ☜ امنیت پیشرو!

عدم امکان رمزگشایی پیامها حتی با داشتن کلید خصوصی



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000	client	server	TLSv1.2	186	Client Hello
2	0.427	server	client	TLSv1.2	1414	Server Hello, Certificate
3	0.428	server	client	TLSv1.2	330	Server Key Exchange
4	0.451	client	server	TLSv1.2	396	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
5	0.870	server	client	TLSv1.2	336	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
6	0.871	client	server	TLSv1.2	395	Application Data
7	1.287	server	client	TLSv1.2	853	Application Data, Application Data, Application Data
8	1.349	client	server	TLSv1.2	395	Application Data
9	1.764	server	client	TLSv1.2	656	Application Data, Application Data
10	1.766	client	server	TLSv1.2	395	Application Data
11	2.181	server	client	TLSv1.2	656	Application Data, Application Data

- **راهکار:** اگر بتوان به مرورگر گفت که از Pre-Master Secret کپی بگیرد، قادریم پیامها را حتی بدون داشتن کلید خصوصی کارگزار رمزگشایی کنیم.

☜ چطور؟

کتابخانه NSS و متغیر محیطی

- مرورگرهایی مثل Firefox و Chrome برای رمزنگاری از کتابخانه‌ای به نام NSS که توسط Mozilla توسعه یافته استفاده می‌کنند.

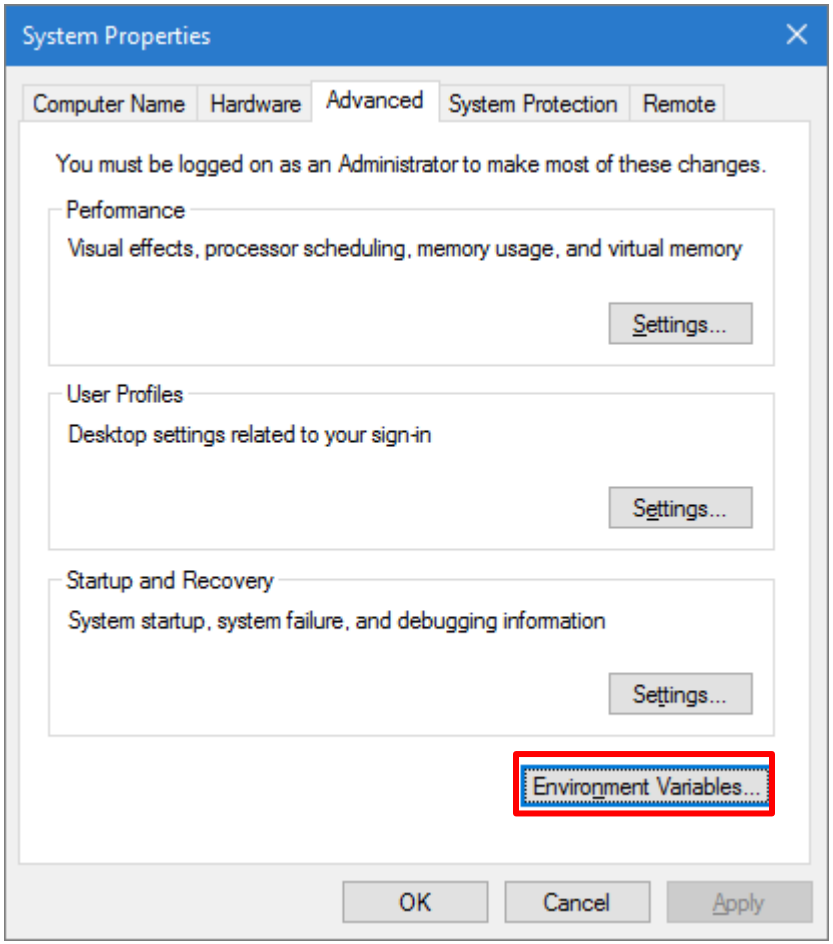
  ☜ کوته‌نوشت Network Security Services

- اطلاعات بیشتر:

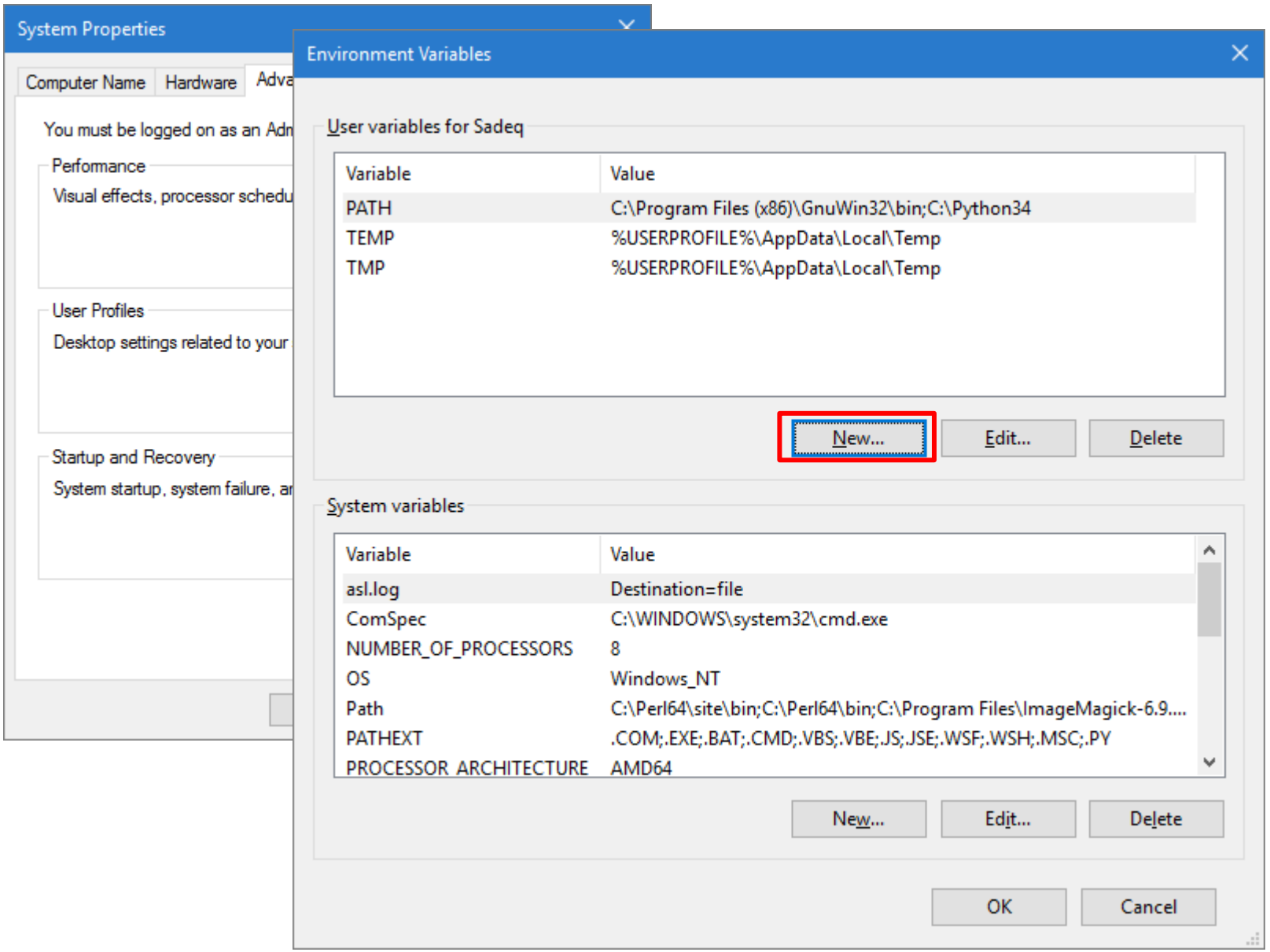
  https://developer.mozilla.org/en-US/docs/Mozilla/Projects/NSS

- می‌توان با تنظیم متغیر محیطی **SSLKEYLOGFILE**، به NSS گفت که Pre-Master Secret را در یک فایل ذخیره کند.

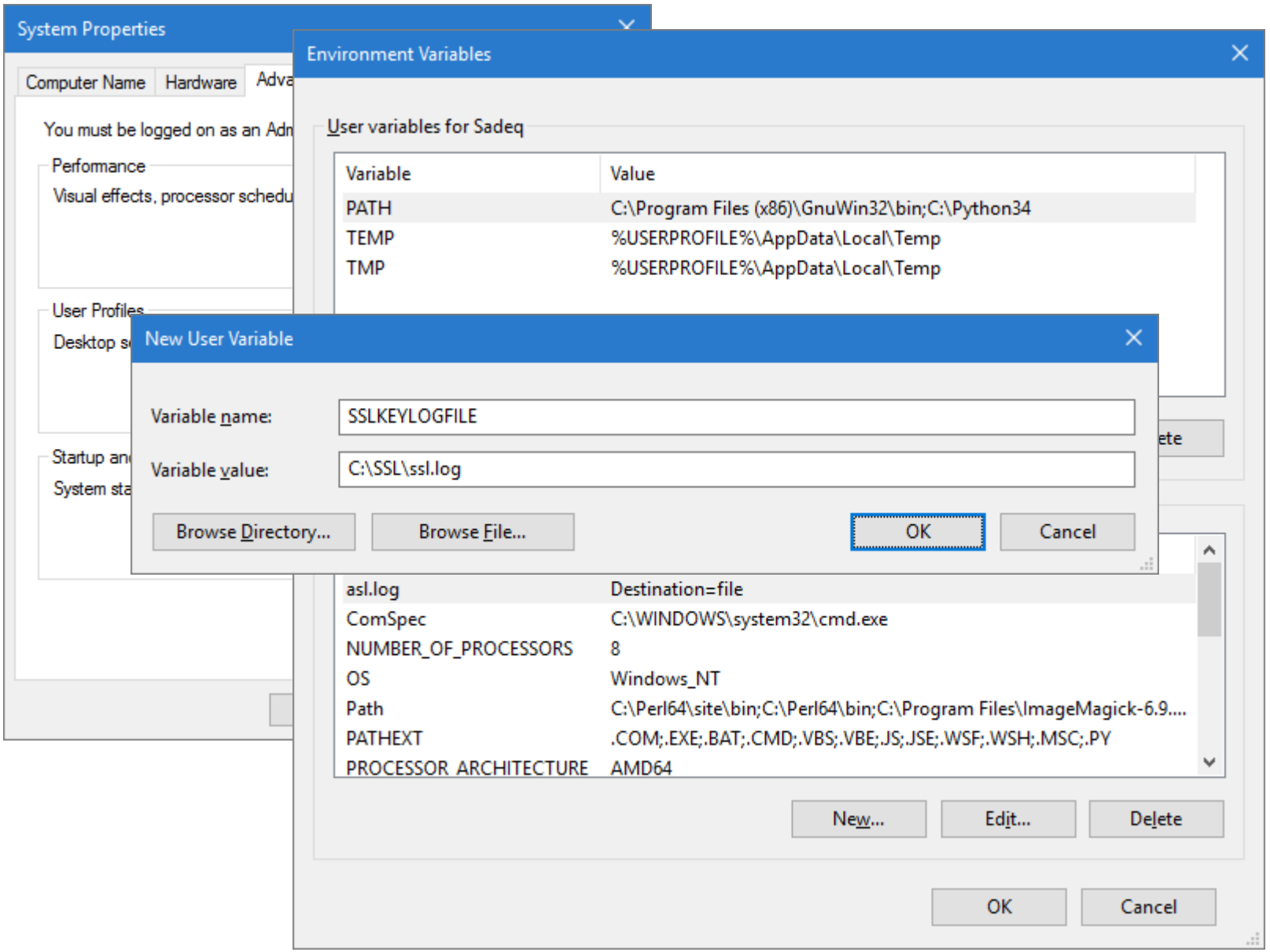
تنظیم متغیر محیطی NSS در ویندوز



تنظیم متغیر محیطی NSS در ویندوز



تنظیم متغیر محیطی NSS در ویندوز



نکات مهم

- پس از تنظیم متغیر محیطی، لازم است مرورگر را بسته و مجدداً باز نمایید.

  ☜ هر برنامه تنها هنگام باز شدن متغیرهای محیطی را می‌خواند.

- لازم است مرورگر به فایل تعیین شده (`ssl.log`) دسترسی write داشته باشد.

  ☜ تنظیم کنترل دسترسی به فایل

- حال مجدداً به صفحه HTTPS مد نظر بروید و تعاملات را با Wireshark ذخیره نمایید.

محتوای فایل ssl.log

```
# SSL/TLS secrets log file, generated by NSS
CLIENT_RANDOM 1723a3ca6b5a… e99d65eeaa5e…
```

- قالب فایل در نشانی زیر مستند شده است:

https://developer.mozilla.org/en-US/docs/Mozilla/Projects/NSS/Key_Log_Format

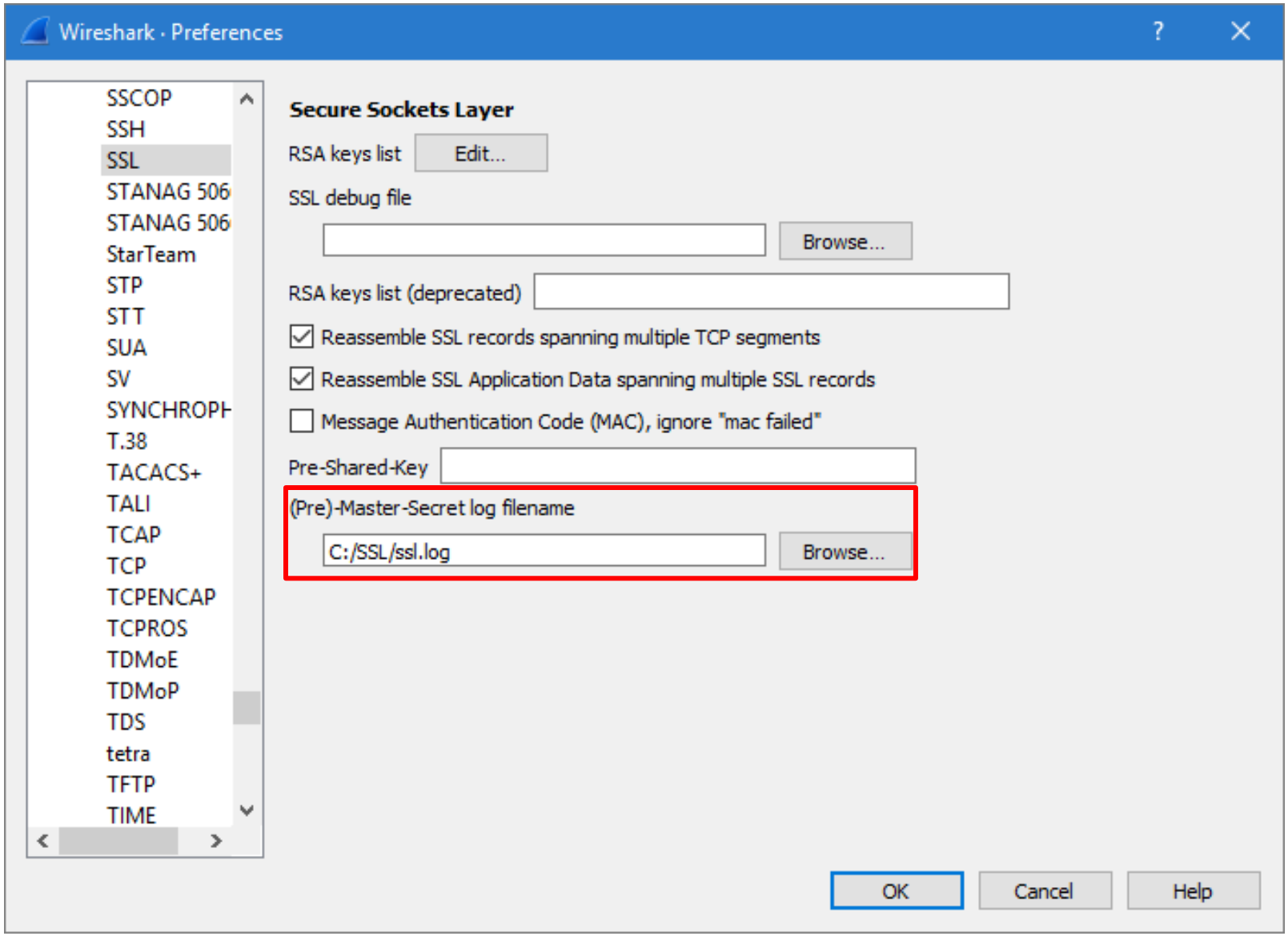
- هر خط از فایل، یا با RSA شروع می‌شود (انتقال کلید) یا با **CLIENT_RANDOM** (تبادل کلید DH).

- در حالت دوم، دو عدد بعد از **CLIENT_RANDOM** می‌آید:

  - عدد نخست (۶۴ بایتی): نانس کارخواه

  - عدد دوم (۹۶ بایت): مقدار Pre-Master Secret

تنظیم نشانی فایل ssl.log در Wireshark



قبل و بعد از رمزگشایی

SSL-DHE.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000	client	server	TLSv1.2	186	Client Hello
2	0.427	server	client	TLSv1.2	1414	Server Hello, Certificate
3	0.428	server	client	TLSv1.2	330	Server Key Exchange
4	0.451	client	server	TLSv1.2	396	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
5	0.870	server	client	TLSv1.2	336	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
6	0.871	client	server	TLSv1.2	395	Application Data
7	1.287	server	client	TLSv1.2	853	Application Data, Application Data, Application Data
8	1.349	client	server	TLSv1.2	395	Application Data
9	1.764	server	client	TLSv1.2	656	Application Data, Application Data
10	1.766	client	server	TLSv1.2	395	Application Data
11	2.181	server	client	TLSv1.2	656	Application Data, Application Data

SSL-DHE.pcapng

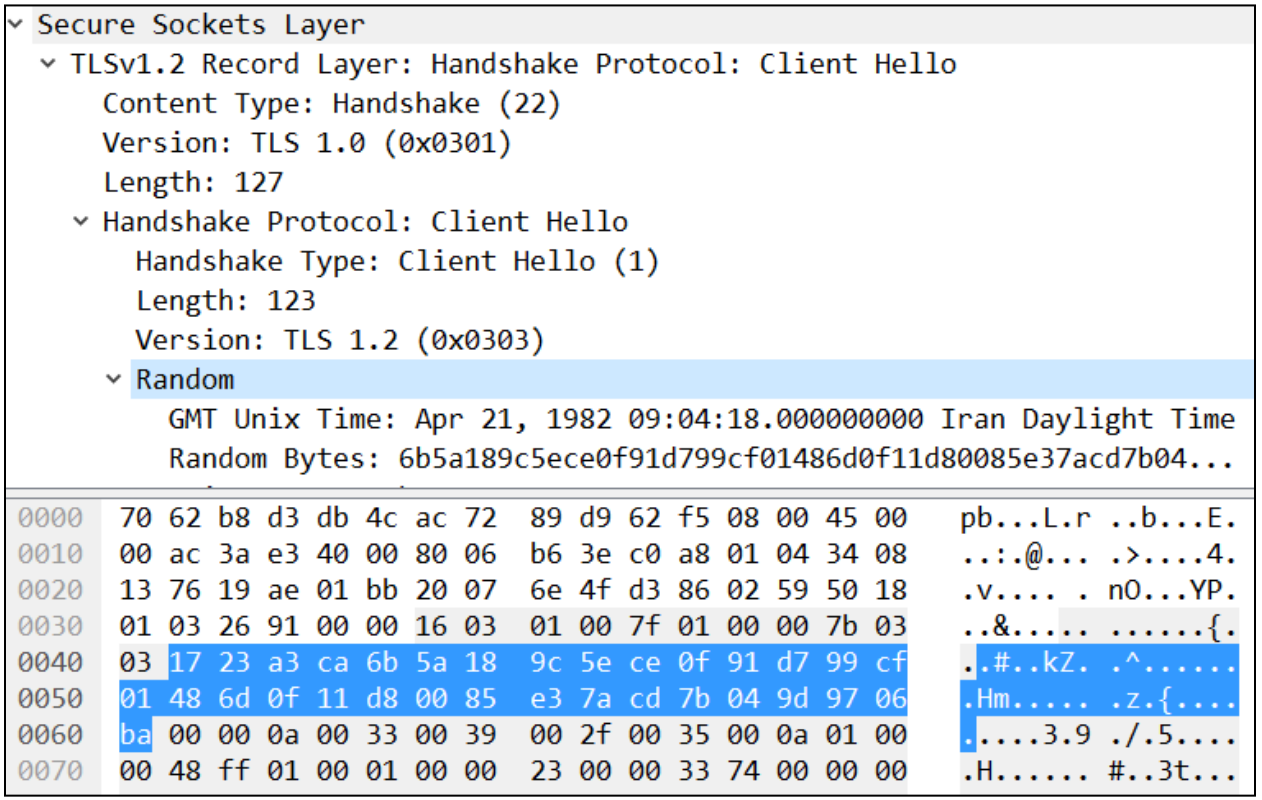
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000	client	server	TLSv1.2	186	Client Hello
2	0.427	server	client	TLSv1.2	1414	Server Hello, Certificate
3	0.428	server	client	TLSv1.2	330	Server Key Exchange
4	0.451	client	server	TLSv1.2	396	Client Key Exchange, Change Cipher Spec, Finished
5	0.870	server	client	TLSv1.2	336	New Session Ticket, Change Cipher Spec, Finished
6	0.871	client	server	HTTP	395	GET / HTTP/1.1
7	1.287	server	client	HTTP	853	HTTP/1.1 200 OK (text/html)HTTP/1.1 200 OK (text/html)
8	1.349	client	server	HTTP	395	GET /favicon.ico HTTP/1.1
9	1.764	server	client	HTTP	656	HTTP/1.1 404 Not Found (text/html)
10	1.766	client	server	HTTP	395	GET /favicon.ico HTTP/1.1
11	2.181	server	client	HTTP	656	HTTP/1.1 404 Not Found (text/html)

یافتن کلید مرتبط به هر نشست

- Wireshark با کمک مقدار Random در پیام Client Hello متوجه می‌شود که از کدام سطر `ssl.log` باید برای رمزگشایی استفاده نماید.



یافتن کلید مرتبط به هر نشست

- Wireshark با کمک مقدار Random در پیام Client Hello متوجه می‌شود که از کدام سطر ssl.log باید برای

```
# SSL/TLS secrets log file, generated by NSS
CLIENT_RANDOM ff638fd766d6ef331b0b9071cdec232cc5ab501c9cabce9406e6ffb408528572 b89
RSA 8f3ea8e1740dccf2 030335fae94f624a361140f31e2ff6a06e95e28b690c1fbc247c3eeb8c11e
CLIENT_RANDOM 84578eaa61939b01c85aebf191bea89d2390095263bbe0da199bb0291bc1b63e 9c2
RSA 0f4b66ea1d8351d0 03030419ce0b7591c6153d29b97b7fafac6891c0b66d7709e0e659cf46635
CLIENT_RANDOM d20a3148378a3fd0eb4e45b9c9b6ef65efe914c4fde1ec362d75d60fbfc889c8 8d6
RSA 3c45cd9a6ed6e522 0303f7fc8874bfda6dc0f8da9f6df5c9cfdf3a7414237216aae64505907f9
CLIENT_RANDOM 245d9191e626856e18eb3f785460c486f75cd29eac203885deff4607a8318e71 a2e
CLIENT_RANDOM 1723a3ca6b5a189c5ece0f91d799cf01486d0f11d80085e37acd7b049d9706ba e99
CLIENT_RANDOM 7d38b2f9a4e949f6919ef572b171ba04a60ae618b4a0af231aac68e7db213313 63b
CLIENT_RANDOM 7a14944cf92f08757314b48c21c90260c340e4bcc4752a10265be8b2e8308cf6 63b
CLIENT_RANDOM a612efbe238f6a3e13b2ee481227861a5b3b9bc67a56152d74ccdc97f159e118 63b
CLIENT_RANDOM ac9bc4336936232cdfcaed6ae42cd278ef0ec97588e993a28e7d3efc294469bd 63b
CLIENT_RANDOM 29942e0ad8a6e8684c0e8ab7a9a7c3dd2327fe0063b31f048b52d9bdf82aa824 63b
```

```
GMT Unix Time: Apr 21, 1982 09:04:18.000000000 Iran Daylight Time
Random Bytes: 6b5a189c5ece0f91d799cf01486d0f11d80085e37acd7b04...
```

```
0000  70 62 b8 d3 db 4c ac 72  89 d9 62 f5 08 00 45 00   pb...L.r ..b...E.
0010  00 ac 3a e3 40 00 80 06  b6 3e c0 a8 01 04 34 08   ..:.@... .>....4.
0020  13 76 19 ae 01 bb 20 07  6e 4f d3 86 02 59 50 18   .v.... . nO...YP.
0030  01 03 26 91 00 00 16 03  01 00 7f 01 00 00 7b 03   ..&..... ......{.
0040  03 17 23 a3 ca 6b 5a 18  9c 5e ce 0f 91 d7 99 cf   .#..kZ. .^......
0050  01 48 6d 0f 11 d8 00 85  e3 7a cd 7b 04 9d 97 06   .Hm..... .z.{....
0060  ba 00 00 0a 00 33 00 39  00 2f 00 35 00 0a 01 00   .....3.9 ./.5....
0070  00 48 ff 01 00 01 00 00  23 00 00 33 74 00 00 00   .H...... #..3t...
```

تبادل الگوریتمهای رمز پس از فعال‌سازی DH

Client Hello

```
Cipher Suites (5 suites)
    Cipher Suite: TLS_DHE_RSA_WITH_AES_128_CBC_SHA (0x0033)
    Cipher Suite: TLS_DHE_RSA_WITH_AES_256_CBC_SHA (0x0039)
    Cipher Suite: TLS_RSA_WITH_AES_128_CBC_SHA (0x002f)
    Cipher Suite: TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
    Cipher Suite: TLS_RSA_WITH_3DES_EDE_CBC_SHA (0x000a)
```

Server Hello

```
    Cipher Suite: TLS_DHE_RSA_WITH_AES_128_CBC_SHA (0x0033)
```

Server Key Exchange

```
Handshake Protocol: Server Key Exchange
  Handshake Type: Server Key Exchange (12)
  Length: 779
  Diffie-Hellman Server Params
    p Length: 256
    p: ffffffffffffffffc90fdaa22168c234c4c6628b80dc1cd1...
    g Length: 1
    g: 02
    Pubkey Length: 256
    Pubkey: 94379a87827107f41f82b41c22c0c15774871772bd664588...
    Signature Hash Algorithm: 0x0401
      Signature Hash Algorithm Hash: SHA256 (4)
      Signature Hash Algorithm Signature: RSA (1)
    Signature Length: 256
    Signature: 7ff7fb88fc9573a68153ffe9ca9cfd3e4314f8145de364fe...
```

Client Key Exchange

```
Diffie-Hellman Client Params
    Pubkey Length: 256
    Pubkey: 9230a4b657fea74a5669af9cc98262892158a6fe29eac2cb...
```

- مقدار Pre-Master Secret برابر کلید تبـادل شـده در DH است.

فهرست مطالب

- معرفی و تاریخچه
- SSL/TLS در سطح بالا
- TLS در عمل
- **جزئیات TLS**
- Heartbleed

تولید master secret (تعریف در RFC 5246)

```
master_secret = PRF(
            pre_master_secret,
            "master secret",
            ClientHello.random ||
            ServerHello.random)
```

- master secret مقداری ۴۸ بایتی است.
- PRF یا تابع شبه تصادفی (Pseudo-Random Function)
  - ☜ **ورودی:** مقداری مخفی، برچسب (label)، و بذر (seed).
  - ☜ **خروجی:** مقداری تصادفی به طول دلخواه.

- در TLS از HMAC برای ساخت PRF استفاده می‌شود.

☜ TLS 1.2، استفاده از SHA-256 یـا بهتـر را بـرای HMAC توصیه می‌کند.

```
P_hash(secret, seed) = HMAC_hash(secret,A(1) || seed) ||
                       HMAC_hash(secret,A(2) || seed) ||
                       HMAC_hash(secret,A(3) || seed) || ...
```

where A() is defined as

A(0) = seed
A(i) = HMAC_hash(secret, A(i − 1))

PRF(secret, label, seed) =
P_hash(secret, label || seed)

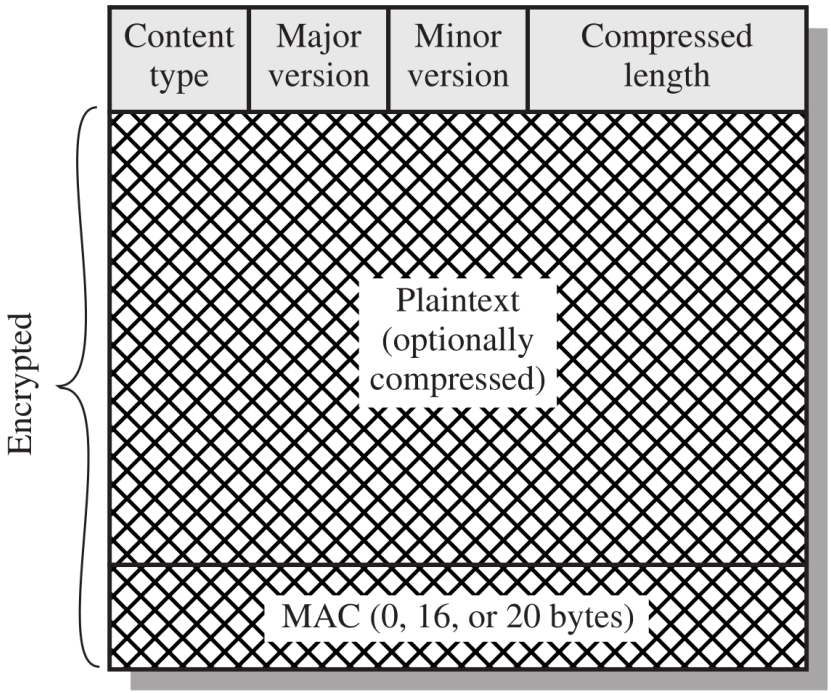
استخراج ۶ کلید نشست

```
key_block = PRF(
                    master_secret,
                    "key expansion",
                    ServerHello.random ||
                    ClientHello.random)
```

- ۶ کلید نشست به ترتیب زیر از روی **key_block** استخراج می‌شوند:
  - Client write MAC secret
  - Server write MAC secret
  - Client write encryption key
  - Server write encryption key
  - Client write encryption IV
  - Server write encryption IV

مثال از کاربرد MAC_write_key

```
MAC = MAC_Algorithm(MAC_write_key,
            seq_num ||
            TLSCompressed.type ||
            TLSCompressed.version ||
            TLSCompressed.length ||
            TLSCompressed.fragment)
```



فهرست مطالب

- معرفی و تاریخچه
- SSL/TLS در سطح بالا
- TLS در عمل
- جزئیات TLS
- Heartbleed

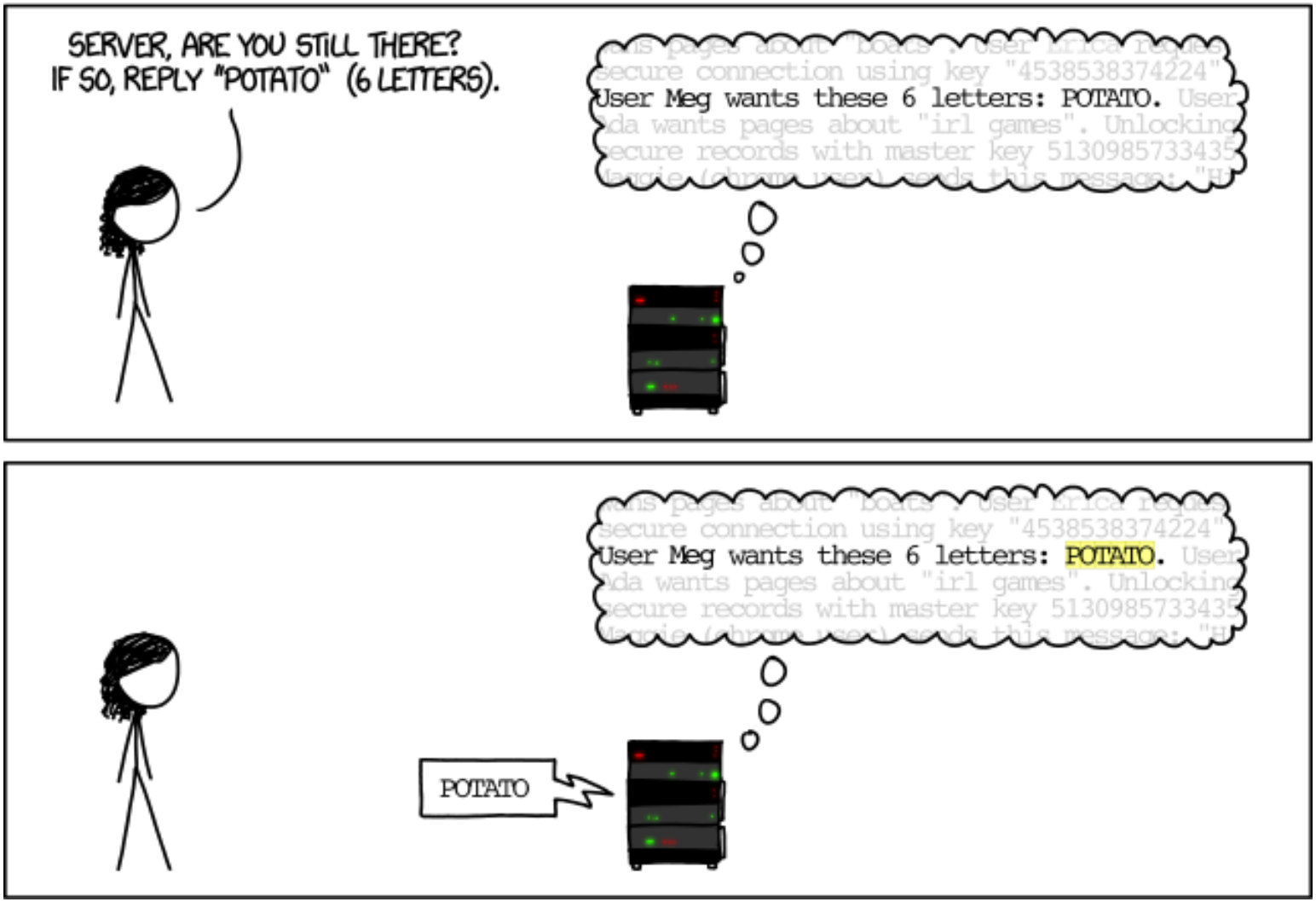
آسیب‌پذیری خونریزی قلبی (Heartbleed)

- یک آسیب‌پذیری بسیار معروف که در سال ۲۰۱۴ در نرم‌افزار OpenSSL کشف شد.
- تأثیر روی میلیونها کارگزار HTTPS در دنیا!
- آسیب‌پذیری در پیاده‌سازی پروتکل Heartbeat؛ توسعه‌ای از TLS تعریف شده در RFC 6520.

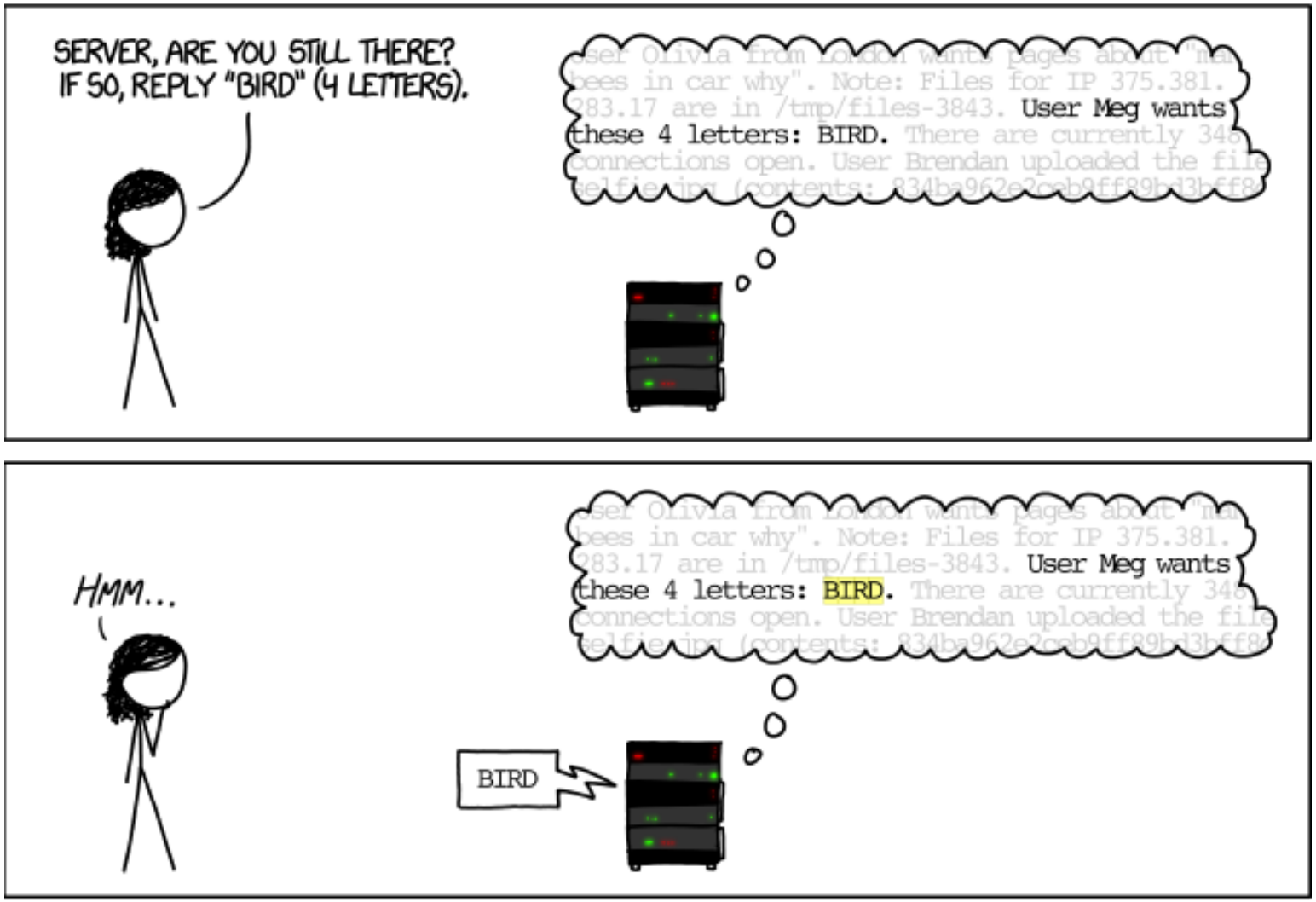
☜ این پروتکل برای بررسی و زنده نگهداشتن اتصال است.

- آسیب‌پذیری از نوع سرریز بافر است.
- از ۲۰۱۱ در کد OpenSSL وجود داشته است.

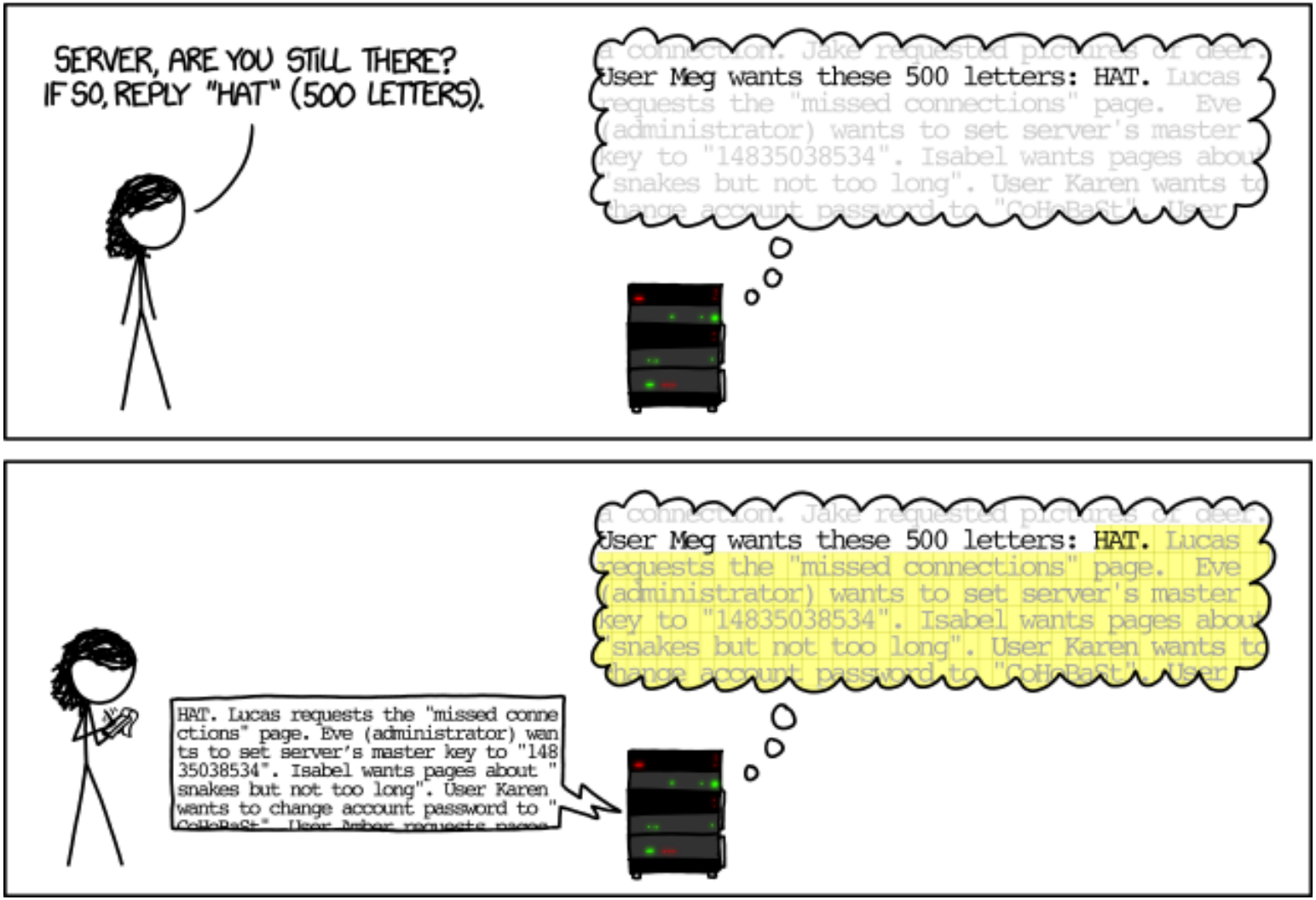
چگونگی سوء استفاده از آسیب‌پذیری Heartbleed



چگونگی سوء استفاده از آسیب‌پذیری Heartbleed



چگونگی سوء استفاده از آسیب‌پذیری Heartbleed



پایان

صفحه درس:

http://ce.sharif.edu/courses/94-95/1/ce442-1/

مراجعه حضوری جهت رفع اشکال: شنبه‌ها ۱۵ الی ۱۶

(طبقه پنجم دانشکده، درب شیشه‌ای جنب آسانسور)

یا در زمانهای دیگر با قرار قبلی

یا به وسیله رایانامه: dousti@ce